

Politechnika Warszawska
Wydział MEiL



Praca Magisterska

Temat: Symulacja komputerowa mechaniki fortepianu.

Autor: Michał Janczak

Nr Albumu: 168599

Prowadzący: dr Cezary Rzymkowski

Warszawa, sierpień 2002r.

„Piękno dzieła sztuki pozostanie zawsze nieodgadnione, to znaczy, że nie będzie można nigdy dokładnie sprawdzić, j a k t o j e s t z r o b i o n e ”.

(Claude Debussy, „Revue SIM” 1913)

1. Wstęp.

Celem niniejszej pracy było przeprowadzenie symulacji komputerowej mechaniki fortepianu oraz weryfikacja jej dokładności. Nie ulega wątpliwości, że w przypadku tak dalece złożonego mechanizmu, jakim jest fortepian, podobna symulacja może mieć charakter jedynie szacunkowy, jako że sama weryfikacja modelu ma również charakter szacunkowy. W niniejszej pracy interesuje nas szeroko rozumiana symulacja fortepianu, a nie jakiegoś konkretnego modelu, czy wręcz egzemplarza. Trzeba mieć bowiem świadomość, że nawet pomiędzy instrumentami tej samej marki zawsze pojawią się drobne różnice w wykonaniu bardzo precyzyjnych części i rozbieżności w zakresie regulacji samego mechanizmu. Dlatego też dążenie do tego, aby wyniki z symulacji i pomiarów eksperymentalnych za wszelką cenę się pokrywały, nie ma większego sensu. Pragniemy jedynie stwierdzić, czy symulacja faktycznie uwzględnia występujące w układzie zjawiska. Weryfikacja ma więc tu zdecydowanie charakter jakościowy, a nie ilościowy. Dlatego też niezbędne do jej przeprowadzenia pomiary eksperymentalne wykonałem nie na prawdziwym instrumencie, ale na modelu jednego klawisza wykonanym w celach dydaktycznych. Naturalnie jego charakterystyki różnią się w pewnym stopniu od charakterystyk prawdziwego fortepianu, jednakże na poziomie dokładności moich pomiarów różnice te są raczej niemierzalne i nie mają dla nas większego znaczenia.

Oddzielną kwestią pozostaje ocena jakości symulacji z punktu widzenia pianisty. Przeprowadzenie jej jednak bez zbudowania sterowanej komputerowo klawiatury wydaje się niemożliwe. Ocena jakości fortepianów jest zagadnieniem dalece subiektywnym mającym więcej wspólnego z psychologią muzyki, aniżeli z mechaniką – zatem opieranie się tu jedynie na matematycznych kalkulacjach nie ma większego sensu. Pojawia się w tym miejscu wyraźna granica pomiędzy sztuką, a nauką, której ta ostatnia przekroczyć nie jest w stanie. Ostateczna ocena możliwości instrumentu zawsze należeć będzie do pianisty, a nie naukowca.

Nie oznacza to jednak, że nauka nie ma nic do zaoferowania muzykowi. Przykładowo przeprowadzone przez wielu naukowców analizy biomechaniczne ruchów ręki podczas gry na fortepianie pozwoliły na lepsze zrozumienie zjawisk i aspektów bezpośrednio związanych z techniką gry takich, jak zmęczenie mięśni, koncentracja, czy precyzja ruchów [4]. W zasadzie bardzo trudno stwierdzić, czy podobne badania miały do tej pory jakiś znaczący wpływ na budowę samych instrumentów. Choć niemal każda większa fabryka fortepianów prowadzi

1. Wstęp.

badania nad udoskonaleniem mechanizmu i poprawą właściwości akustycznych, od ponad stu lat budowa fortepianów praktycznie nie uległa zmianie! Zgoła inaczej rzec się ma, jeśli chodzi o instrumenty elektroniczne. Tutaj nieustannie pojawiają się nowe technologie mające sprawić, że pianino elektryczne będzie pod względem akustyki i mechaniki nie do odróżnienia od tradycyjnego instrumentu. W przypadku instrumentów wykorzystujących gotowe *sample* barwa dźwięku jest bardzo zbliżona do oryginalnej. Znacznie więcej kłopotów przedstawia odtworzenie właściwości mechaniki. Współczesne klawiatury są wprawdzie wyważane poprzez podwieszoną pod klawiszem ruchomą przeciwwagę. Posiadają nawet możliwość regulacji oporów klawiszy, jednak dla zawodowego pianisty okazują się raczej bezużyteczne. Nie odwzorowują one zjawiska podwójnej repetycji, wpływu tłumika i szumów klawiatury na barwę dźwięku [9] oraz szeregu innych zjawisk, z których zapewne nie wszystkich jesteśmy świadomi! Taki stan rzeczy wynika w dużej mierze ze specyfiki rynku, na jaki trafiają podobne instrumenty. Dla zdecydowanej większości odbiorców powyższe „szczegóły” nie mają większego znaczenia i prawdopodobnie dlatego nie opłaca się budować na szeroką skalę instrumentów wiernie oddających właściwości mechaniki.

Jednym z celów przeprowadzonej w niniejszej pracy symulacji było rozeznanie się co do możliwości zbudowania klawiatury z aktywnym wyważeniem, tzn. sterowanym przez komputer, co pozwoliłoby na dokładniejsze odwzorowanie prawdziwego instrumentu. Pojawia się tu szereg problemów. Przede wszystkim należało zbudować na tyle prosty model matematyczny, aby mógł być on przeliczany w czasie rzeczywistym, a następnie za pośrednictwem siłowników- najrozsądzniejszymi wydają się elektromagnetyczne- sterować mechanizmem klawiatury i jednocześnie syntezować dźwięk uwzględniając przy tym tłumiący wpływ tłumików i młoteczka oraz różnego rodzaju szумy i inne zjawiska akustyczne. Zagadnienie co najmniej niebanalne.

Zanim przejdziemy do szczegółów symulacji, warto nieco lepiej przyjrzeć się temu, co było symulowane.

2. Historia fortepianu.

Historia fortepianu łączy się z rozwojem dwóch innych instrumentów: klawesynu i klawikordu [13]. Łączyła je wspólna cecha - klawisze, dotychczas stosowane głównie przy budowie organów. Łacińska nazwa *clavis* odpowiada polskiej „klucz”, w organach bowiem klawisze-klucze „otwierały” powietrzu dostęp do piszczalek. W klawikordzie i w klawesynie miały inne zadanie. Nie można jednak omawiać wspólnie tych tak bardzo różnych instrumentów!

Klawesyn wszedł w użycie później niż klawikord i pierwsi Niemcy dali mu nazwę, która odpowiadała kształtowi instrumentu: *Flugel*, co znaczy po polsku „skrzydło”. Pudło klawesynu bowiem miało już - tak jak u dzisiejszego fortepianu - kształt ptasiego skrzydła. W pudle owym napięte były prostopadle do klawiatury struny - basowe z mosiądu najdłuższe i wiolinowe z żelaza coraz krótsze. Dźwięk ze strun dobywano w ten sposób: naciśnięcie klawisza powodowało skok w górę umieszczonego po drugiej stronie klawiszowej dźwigni drewnianego klocka zwanego skoczkiem; do klocka przymocowane było zastrzone pióro orle, sokole albo krucze; piórko to zarywało strunę, z której dobywał się dźwięk brzęczący dość głośno, ale zawsze jednakowy w nasileniu.

Inaczej rzecz się miała z klawikordem. Ten budowano na kształt prostopadłościanu, w którego dłuższą ściankę wmontowana była klawiatura, a struny napinano równolegle do niej. Skoro naciskało się klawisz, umieszczony po drugiej stronie klawiszowej dźwigni metalowy język podrywał się i bił w strunę, wywołując dźwięk mało donośny, ale możliwy do cieniowania słabszym lub silniejszym uderzeniem w klawisz.

Jeszcze był spinet, najbardziej popularny w barokowych i rokokowych salonach. Pudło i układ strun miał takie jak w klawikordzie, lecz struny zarywane piórkiem jak w klawesynie. Małe spinety budowano często bez nóg, instrument wnoszony był przez lokaja, który kładł go na stoliku, po czym przy klawiaturze siadała wytworna dama i towarzyszyła sobie grą do tkliwych pieśni pasterskich, śpiewanych ku zachwytniowi wielbicieli.

Klawesyn więc miał dźwięk silny, stosowny do współprzmienia z orkiestrą, ale niepodatny do dynamicznego cieniowania. Klawikord, pozwalający na cieniowanie brzmienia, dźwięczał za to tak nikle, że używany mógł być tylko jako instrument solowy w niewielkich pomieszczeniach. Konstruktorzy spędzali bezsenne noce: jak połączyć te dwie

2. Historia fortepianu.

zalety - silnego brzmienia i możliwości cieniowania (czyli ekspresji) - w jednym, doskonalszym instrumencie?

Na właściwy pomysł wpadł Włoch, Bartolomeo Cristofori z Padwy, który w 1711 roku zastąpił piórka i metalowe językczki młotkami uderzającymi o struny. Ponadto (rzecząc bodaj jeszcze ważniejsza!) uniezależnił młotki od klawiszowych dźwigni. Natychmiast po uderzeniu w strunę - obojętnie, czy przyciskało się dalej klawisz, czy puszczało - młotek odskakiwał od struny, która wibrowała swobodnie, a on przygotowany już był do następnego uderzenia.

Jak to przeważnie bywało z wszelkiego rodzaju innowacjami, także i ta nie od razu została przyjęta przez współczesnych. Pierwszym przeciwnikiem fortepianu Cristoforiego - zanim się do niego nie przekonał - był ... sam wielki Jan Sebastian Bach... Ano... Pomyłki w sądach zdarzają się i geniuszom! Nie mniej wielki w swojej literackiej dziedzinie Voltaire broniąc zagrożonego klawesynu nazwał fortepian „instrumentem garkotników”.

Zaniepokojeni konstruktorzy klawesynów rzucili się do tym energiczniejszego udoskonalania swoich fabrykatów. Zaczęły powstawać kolejno: klawesyn „orchestrina” imitujący brzmieniem kwartet smyczkowy; klawesyn zbudowany przez niejakiego Blahę, a naśladowany organy, bęben, kastanietę... grzmoty i deszcz; klawesyn „harmoniczny” naśladowujący instrumenty perkusyjne i dęte; klawesyn „anielski” o strunach nie szarpanych piórkiem, lecz pocieranych filcem i aksamitem. Jeszcze Mozart, a nawet - w początkach swej kariery - Beethoven grywali na klawesynie, tak dzielnie bronił się zagładzie ustępujący ze światowej widowni staruszku.

Skąd wzięła się nazwa „pianoforte” nadana jego młodemu przeciwnikowi? Czy wraz z nowym mechanizmem została wynaleziona przez samego Cristoforiego? Nie! Narodziła się z zachwytu pierwszych słuchaczy, którzy podziwiali, że na tym pięknym instrumencie można grać i cicho (po włosku piano), i głośno (forte), i te dwa okrzyki podziwu złączyli w jedno. Znani z przekory Polacy nazwę czym przedzej odwrócili i dali instrumentowi miano „fortepian”.

Obserwując dalsze losy fortepianu, zauważymy, iż o jego rozwoju decydowało współdziałanie elementów, zdawałoby się, zupełnie od siebie niezależnych: mózgu, natchnienia i zmysłu handlowego. Skoro konstruktor wpadał na jakiś pomysł udoskonalenia fortepianowej mechaniki, zachęcony kompozytor natychmiast uwzględniał to w swej twórczości. Wirtuozi sprawdzali teraz z kolei, czy mechanika nowych instrumentów

2. Historia fortepianu.

rzeczywiście ułatwia wykonanie nowych kompozycji. Dochodził wreszcie handlowiec z własnymi spostrzeżeniami: jakie odmiany mechanizmów „biorą” na rynku zbytu, a jakie nie, i o co dobrze byłoby się jeszcze postarać.

Dopingując się wzajemnie, konstruktor, kompozytor, wirtuoż i handlowiec wspólnie przyczyniali się do coraz szybszego, choć przecie niełatwego doskonalenia fortepianu.

Jeszcze w połowie XVIII wieku pierwszy na większą skalę fabrykant fortepianów, Niemiec Gotfryd Silbermann, ulepszył mechanizm Cristoforiego tak dalece, że zdołał przełamać niechęć Jana Sebastiana Bacha. „Ten instrument już mi odpowiada” - mruknął Bach ukontentowany. Pochwała owa rozeszła się błyskawicznie i wkrótce Silbermann otrzymał zamówienie na kilka fortepianów od króla pruskiego Fryderyka II. To zadecydowało o przyszłości i fabrykanta, i jego nowego fabrykatu.

W 1774 roku francuski konstruktor Merlin daje fortepianowi lewy pedał, który przesuwa cały mechanizm młoteczkowy nieco w prawo, co powoduje cichsze brzmienie instrumentu.

W tymże roku u brzegów Ameryki rozbija się żaglowiec „Pedro”. W Nowym Jorku odbywa się licytacja uratowanych z pokładu ruchomości, między innymi fortepianu, który ląduje jako pierwszy fortepian na nowej ziemi. Rzecz będzie brzemienna w skutki! W 1778 roku Anglik Bury obciąża fortepianowe młotki filcem na miejsce dawnej skóry. To zmienia charakter i barwę dźwięku. Ton fortepianu matowieje, ale zarazem staje się krągły i soczysty.

W 1797 roku w górach Harzu, w biednej rodzinie obarczonej aż dwanaściorgiem dzieci, przychodzi na świat chłopiec, zapisany w księgach metrykalnych jako Henryk Engelhardt Steinweg. Kiedy wojska napoleońskie, a wśród nich także i pułki niemieckie, przygotowują się do tragicznej bitwy pod Waterloo, szeregowiec Steinweg - dla podniesienia serc kolegów - robi z deski i kilku drutów cytrę i gra na niej. Po czterdziestu latach tenże Steinweg znajdzie się w Ameryce, zmieni nazwisko na Henry E. Steinway i będzie twórcą po dziś pierwszej w świecie fabryki fortepianów.

W 1811 roku angielski konstruktor Robert Wornum montuje struny fortepianu pionowo i konstruuje pierwsze „pianino”, nazwane tak dla mniejszych rozmiarów i mniej donośnego dźwięku.

W 1816 roku Beethoven wymusza na wiedeńskim fabrykancie Streicherze rozszerzenie klawiatury fortepianowej do sześciu i pół oktawy. Na mniejszej skali dusił się ze swymi potężnymi sonatami.

2. Historia fortepianu.

Rok 1823 jest w historii rozwoju fortepianu przełomowy. Sławny francuski konstruktor Sebastian Erard (twórca nowoczesnej harfy) dodaje mechanizmowi młoteczkowemu w fortepianie system repetycyjny. Dotychczas młoteczek po uderzeniu w strunę odpadał aż do swej wyjściowej pozycji. Erard skomplikował mechanizm łączący klawisz z młoteczkiem w ten sposób, że jeśli przytrzymywało się palcem klawisz, to młoteczek po uderzeniu wprawdzie odpadał, lecz zostawał w najbliższej odległości od struny, gotowy do bardzo szybkiego powtórzenia uderzenia, czyli repetycji. Zdawałoby się drobny, a przecież niezmiernej doniosłości wynalazek Erarda dał grającym na fortepianie nowe możliwości użycia instrumentu. Z jednej strony pozwolił na miękkie a dokładne łączenie jednego dźwięku z drugim, czyli na grę legato, bez której nie do pomyślenia byłyby najpiękniejsze karty twórczości Chopina. Z drugiej - repetycja Erardowska zezwoliła na bardzo szybkie powtarzanie jednego dźwięku, bez czego znów nie mógłby powstać taki na przykład klasyczny utwór wirtuozowski jak „Campanella” Liszta. Techniczne podstawy subtelnie uduchowionej gry i zarazem najbliższego popisu - oto co przygotował Erard dorastającym tymczasem Chopinowi i Lisztowi.

W 1825 roku drewniana rama, na której napinane były struny fortepianowe, zastąpiona zostaje przez Amerykanina Babcocka ramą z lanego żelaza. Trochę wcześniej Erard (nie Sebastian, lecz inny, pomniejszy) wzmacnił ramę przeciągniętymi wzduż strun, od jednego brzegu ramy do drugiego, metalowymi żebrami. Trzeba budować fortepiany koncertowe coraz większe, o coraz potężniejszym brzmieniu. Siła naciągu strun w instrumentach z 1810 roku równa była ciężarowi około 5000 kilogramów. Siła naciągu w dzisiejszych fortepianach odpowiada ciężarowi około 40 000 kilogramów!

Wreszcie, w 1830 roku, Niemiec Jan Pape wpadł na pomysł skrzyżowania w fortepianie strun basowych z wiolinowymi. Taki układ pozwolił na napinanie strun odpowiedniej długości skośnie w instrumentach o niezbyt wydłużonym kształcie. Mniej więcej w tym samym czasie ujednolicony zostaje typ strun: najniższe basowe są pojedyncze, ze stali, owinięte drutem mosiężnym; wyższe basowe - podwójne i też owinięte drutem mosiężnym; wiolinowe natomiast są potrójne i bez żadnego owinięcia. Potrójne - to znaczy, że młoteczek uderza jednocześnie w trzy struny o tej samej wysokości dźwięku, co daje brzmienie potrójnej siły i nośności.

Tak to przez wiek cały, wysiłkiem wielu ludzi, fortepian doskonalił się, aby stać się tym nieporównanie szlachetnym instrumentem, jakim jest dzisiaj.

2. Historia fortepianu.

Są cztery typy fortepianów: krótki gabinetowy, salonowy - zwany także półkoncertowym, koncertowy o długości 2 metrów 60 centymetrów i wielki koncertowy długości 3 metrów 10 centymetrów. Do niedawna tylko instrumenty koncertowe miały skalę dźwiękową o szerokości siedmiu i pół oktawy, dzisiaj skalę taką miewają i pianina. Do konstrukcji wnętrza instrumentu dobiera się drewno spośród dwustu różnych gatunków, specjalnie obrabianych, preparowanych i suszonych.

W naszym stuleciu najdoskonalsze fortepiany i pianina produkowano w firmach: Pleyel (Francja), Bechstein, Bluthner (Niemcy) oraz Steinway (Ameryka). Ostatnio do konkurencji przystąpiły firmy japońskie - Yamaha i Kawai. „Steinway” ma najpotężniejsze i najdalej niosące brzmienie, ale i najoporniejszą klawiaturę. „Bechstein” jest nieco lżejszy. Najlżejsze zaś – „Pleyel” i „Bluthner”, więc stosowne dla gry kobiet, przy czym „Bluthner” ma brzmienie jasne, krystaliczne, dobrze nadające się do interpretowania utworów błyskotliwych, lecz mniej na przykład do sonat Beethovena.

Wszystko jest zresztą kwestią gustu. Beethoven swą „Appassionate” komponował na fortepianie Broadwooda. Liszt grywał wyłącznie na instrumentach Erarda. Chopin zaś, gdy koncertował w Anglii, miał w swoim pokoju do ćwiczeń trzy fortepiany: Pleyela, Erarda i Broadwooda. W zależności od tego, jak się czuł danego dnia z siłami używała fortepianu cięższego lub lżejszego.

Nasuwa się ostatnie pytanie: czy i jakich dalszych technicznych udoskonaleń należy spodziewać się w mechanice fortepianu? Raczej żadnych, skoro od mniej więcej wieku mechanika ta już się zasadniczo nie zmienia, a wiarę w trwałą niezmiennosć dzisiejszych modeli fortepianów utwierdzić w nas mogą właśnie próby rozbudowania tych modeli. Ani fortepian o dwóch klawiaturach, co miało skrócić rzuty palców między odległymi klawiszami, ani elektryczna pianola, która groziła zmienieniem fortepianu w katarynkę, ani nawet interesujący w pomyśle fortepian ćwierćtonowy konstrukcji Czecha Aloisa Haby - żadna z tych nowalijek nie utrzymała się dłużej. I mechanizmy instrumentów bowiem muszą stosować się do praw sztuki, które mówią, że doskonałości nie wolno przerysowywać.

3. Model matematyczny.

Zamodelowanie mechanizmu fortepianu jest zagadnieniem złożonym. Pojawia się tu konieczność symulowania zderzeń o charakterze niesprężystym i związanej z tym dyssypacji energii. Innym problemem jest modelowanie zjawiska tarcia pomiędzy ruchomymi elementami. W układzie występują dwuelementowe łańcuchy kinematyczne nastręczające trudności natury numerycznej trudne do ominięcia. Wreszcie kwestia absolutnie kluczowa – szybkość algorytmu, która w zasadzie determinuje przydatność całej simulacji, jeżeli chcielibyśmy wykorzystać ją do sterowania instrumentem elektronicznym. Pojawia się tu dylemat dotyczący całościowego podejścia do zagadnienia. Istnieją dwa możliwe postępowania. Pierwsza to zbudowanie wiernego modelu w oparciu o prawa mechaniki, zapewniającego dużą dokładność wyników, ale przez swoją nieuniknioną złożoność kosztownego, jeśli chodzi o czas obliczeń. Alternatywą jest podejście czysto inżynierskie – zbudowanie bardzo uproszczonego modelu zawierającego w sobie różnego rodzaju parametry dobierane empirycznie i nie mające żadnego pokrycia w prawach mechaniki zapewniające jednak możliwie dużą zbieżność z wynikami z pierwszego modelu. Taki algorytm działałby nieporównywalnie szybciej umożliwiając obliczenia w „czasie rzeczywistym”, a tym samym zastosowanie w instrumencie.

W niniejszej pracy przedstawione są oba podejścia do problemu- model „dokładny” i „uproszczony”. Pierwszy algorytm uruchomiony na standardowym komputerze PC (z zegarem 1,2 GHz) okazał się stanowczo za wolny (ok. 1000 razy!), aby można go było wykorzystać w sterowaniu klawiaturą. Posiada jednak dużą wartość, jeśli chodzi o analizę dynamiki całego układu niezbędną do zbudowania modelu „uproszczonego”. Pozwala ocenić wpływ poszczególnych zjawisk na działanie całości mechanizmu.

Zbudowanie modelu uproszczonego wiąże się z koniecznością przeprowadzenia bardzo żmudnych badań empirycznych- najlepiej na fizycznym mechanizmie sterowanym przez komputer. Nie wszystko bowiem da się wyrazić w formie wykresów i liczb. W pracy nad optymalnym modelem klawiatury osobą niezbędną obok inżyniera byłby pianista, a dokładniej cały zespół pianistów oceniających zalety i wady poszczególnych modyfikacji algorytmu. Naturalnie podobnej weryfikacji mogliby oni dokonać jedynie przy wykorzystaniu prototypu sterowanej numerycznie klawiatury, której wykonanie stanowi zupełnie oddzielny problem. W mojej pracy przedstawiłem więc nie gotowy model nadający się do wykorzystania w przemyśle, ale jedynie bardzo wstępne jego opracowanie, które w

3. Model matematyczny.

połączeniu z opisanyem szczegółowo modelem „dokładnym” stanowi pierwszy krok na drodze budowy instrumentu nowej generacji.

3.1. Budowa mechanizmu młoteczkowego

Zanim przejdę do szczegółów symulacji kryjących w sobie wyżej wspomniane trudności omówię ogólną budowę mechanizmu i zasadnicze założenia modelu. Mechanizm fortepianu (rys.1) składa się z następujących zespołów: klawiszowego, figurowego, młotkowego i tłumikowego.

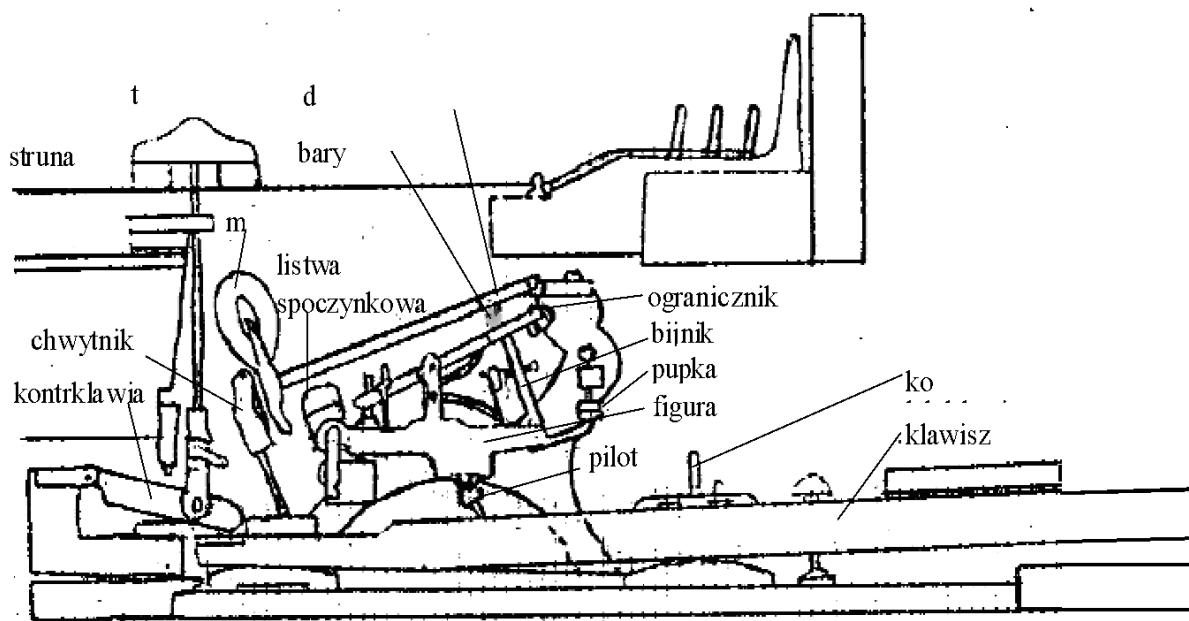
Zespół klawiszowy składa się z klawisza osadzonego na dwóch pionowych kołkach zapewniających ruch tylko w płaszczyźnie pionowej. Leżący w połowie długości klawisza kołek osiowy wyznacza oś obrotu klawisza. Tylnie ramię dźwigni podnosi mechanizm tłumika, chwyta młoteczka i za pośrednictwem pilota (krótkiej mosiążnej śruby z jednorodną cylindryczną głowką) całą figurę.

Figura składa się z trzech części: dolnej dźwigni pośredniej, dźwigni repetycyjnej i bijnika. Dźwignia repetycyjna służy podtrzymywaniu bródki młoteczka w zawieszeniu. Osadzona na osi porusza się na podobieństwo wahadła, ale nie może opadać w dół pod ciężarem młoteczka. Nie pozwala na to sprężyna repetycyjna oraz filcowe ograniczniki ruchu. W jej przedniej części zrobiono prostokątne okienko na przejście bijnika, tak, aby mógł się on stykać z bródką młoteczka. Okienko oraz filcowe występuje ograniczają zakres ruchów bijnika.

Zespół młoteczkowy składa się z trzonu i główka, tzn. grubej warstwy filcu oklejonego na rdzeniu. Zakończony jest on ogonkiem do stykania się z chwytnikiem. Od dołu do trzonka przymocowana jest okrągła baryłczka oklejona zamszem- tzw. bródka, o którą opiera się główka bijnika. Młotki opadają na listwę spoczynkową, oklejoną poduszką, przymocowaną końcami do ramy mechaniki. Bijniki są odłączone od bródek wskutek działania występów oporowych (pupek), usytuowanych nad ramionami bijników.

Zasada działania zespołu tłumikowego jest prosta: przy naciśnięciu klawisza koniec dźwigni klawiszowej unosi się do góry i styka z dźwignią kontrklawiatury. Podnosi ją do góry razem z tłumikiem, zwalniając w ten sposób struny. W ruchu powrotnym tłumik, pod ciężarem główka, drutu i figury, opada w dół i tłumi struny.

3. Model matematyczny.

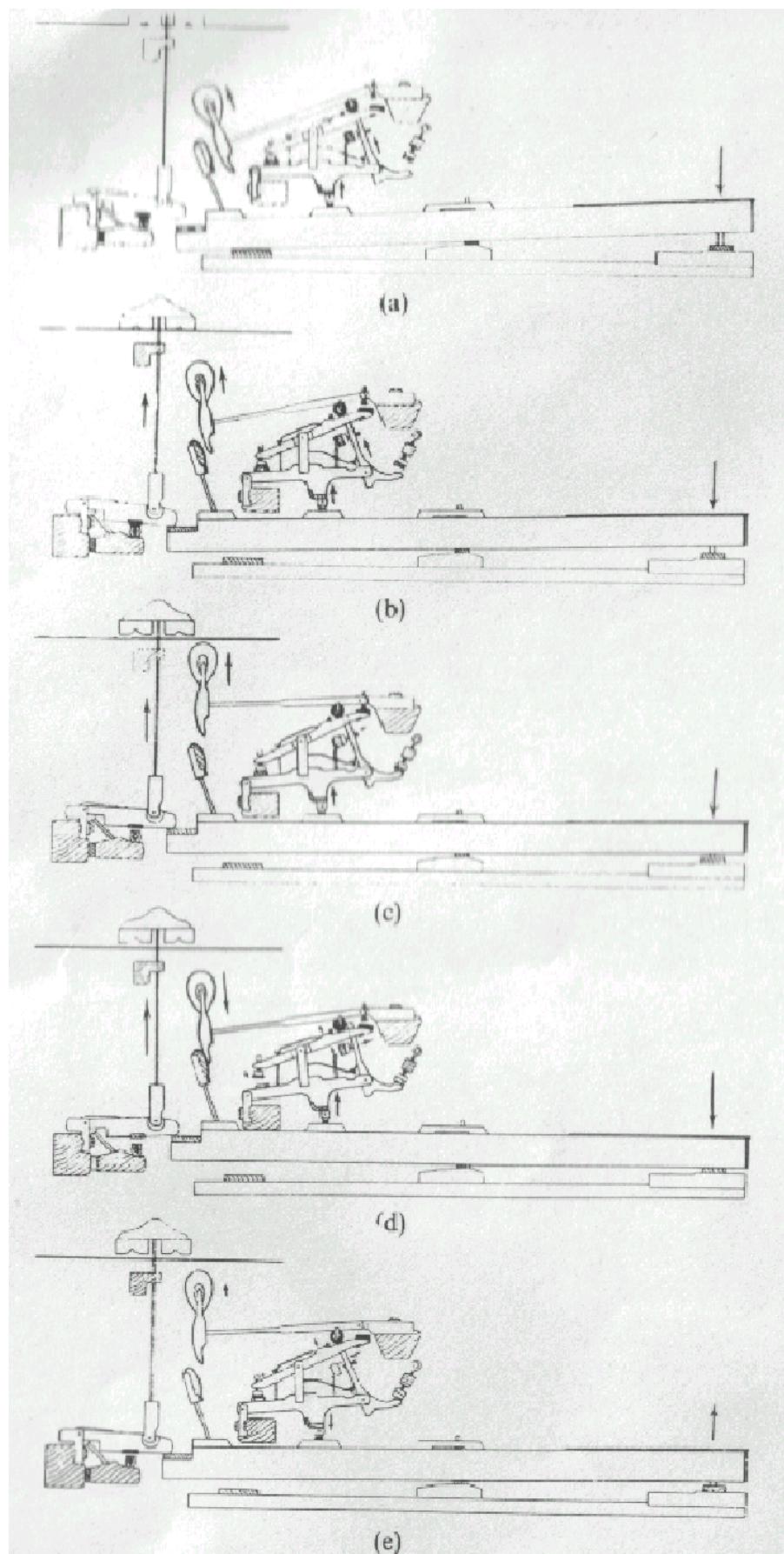


Rys.1. Schemat budowy mechaniki fortepianu marki Stainway.

Idea mechanizmu podwójnej repetycji polega na tym, aby był on gotowy do następnego uderzenia możliwie najszybciej i przy jak najmniejszym ruchu klawisza. Rys.2. pokazuje tę zawiłą sekwencję. Przy uderzeniu w klawisz (a) pilot podnosi figurę, bijnik podrzuca młotek (b), a następnie występ oporowy (pupka) odłącza bijnik od bródki młotka (c). Młotek opadając po uderzeniu w strunę opiera się na dźwigni repetycyjnej i przechyla ją swoim ciężarem będąc jednocześnie wyhamowywanym przez chwytnik(d). Kiedy zwolnimy nieco klawisz tak, aby podniósł się o kilka milimetrów, nastąpi rozdzielenie chwytnika i młotka. W wyniku tego dźwignia repetycyjna uniesie nieco młoteczek, dzięki czemu oswobodzony od występu oporowego bijnik będzie mógł się wsunąć pod bródkę młotka (e). W ten sposób mechanizm, przy prawie całkowicie naciśniętym klawiszem, jest znów gotowy do powtórzenia uderzenia. Taka budowa mechanizmu podwójnej repetycji pozwala powtarzać ten sam dźwięk z dużą częstotliwością przy niewielkich ruchach klawisza. Częstotliwość ta zależy od rodzaju mechaniki i od jakości jej wykonania. W lepszych instrumentach osiąga wartość 12-15 uderzeń na sekundę.

Przy budowie fortepianów zachowuje się pewne optymalne parametry. Dla prawidłowo wyregulowanej mechaniki skok młotka wynosi 46-50 mm, uchylenie, czyli głębokość opadania klawiszy 10 mm (z dokładnością +0,1; -0,05 mm), a stosunek długości przedniego ramienia dźwigni klawiszowej do odcinka pomiędzy kołkiem osiowym a pilotem 1,9-2,1.

3. Model matematyczny.

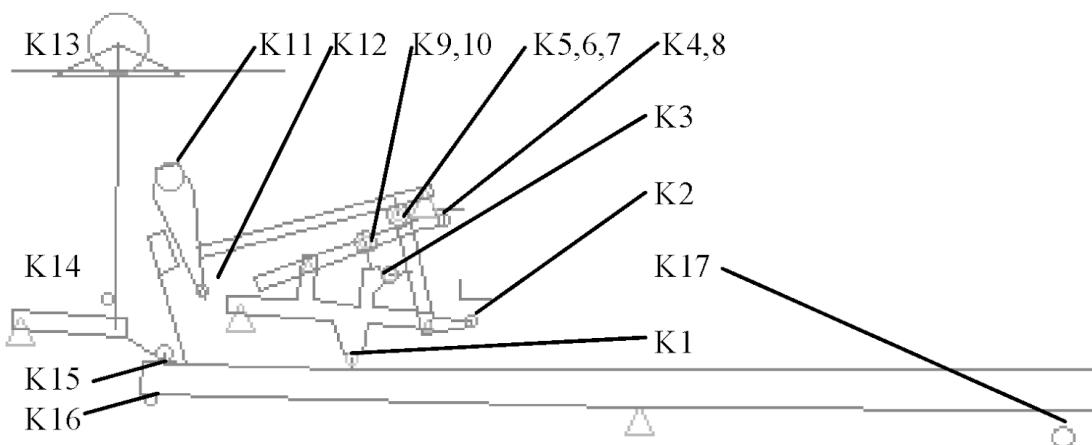


Rys.2. Schemat działania mechaniki fortepianu.

3.2. Założenia symulacji

W mojej symulacji znacznie uprościłem geometrię sprowadzając ją do odcinków prostych i okrągów opisanych w dwóch wymiarach. To w pełni wystarcza do zapewnienia właściwych oddziaływań jednych elementów na drugie. Są one wyszczególnione na rys.3. w formie kólek.

Przyjąłem lewoskrętny układ współrzędnych. Wszystkie kąty mierzone są przeciwnie do zegara. Cały modelowany układ składa się z 6-ciu ruchomych ciał poruszających się ruchem obrotowym lub płaskim będącym złożeniem dwóch ruchów obrotowych oraz umownego ciała "zerowego" zawierającego nieruchome elementy mechanizmu takie, jak: struna, pupka i ogranicznik dźwigni repetycyjnej. Ozym 6-ciu ciałom odpowiadają kolejno dźwignia pośrednia figury (1), dźwignia repetycyjna (2), bijnik (3), młotek (4), tłumik (5) i klawisz (6). Wprawdzie tłumik zbudowany jest z dwóch części – dolnej dźwigni kontrklawiatury i połączonego z nią przegubowo drutu zakończonego filcem tłumikowym. Jednakże „zamrożenie” przegubu i zastąpienie tych elementów jednym sztywnym ciałem nie wprowadza żadnych znaczących zmian w dynamice młoteczka i klawisza, które to interesują nas najbardziej. Dlatego też uproszczenie takie jest dopuszczalne. Położenie i prędkość kątową każdego z ciał traktuję jako zmienne niezależne. Prędkość obrotowa figury jest zarazem prędkością unoszenia dźwigni repetycyjnej i bijnika (występują tu zatem dwuczłonowe łańcuchy kinematyczne).



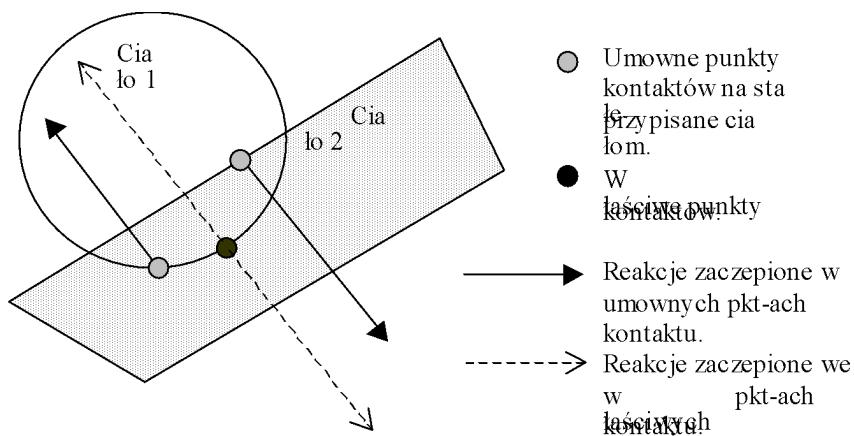
Rys.3. Uproszczony schemat mechanizmu młoteczkowego oraz występujące w nim kontakty. K1- pilot-stopka; K2 - bijnik-pupka; K3 – bijnik-ogranicznik lewy; K4 – bijnik-ogranicznik prawy (w dźwigni rep.); K5 – bijnik (powierzchnia tylna)-baryłka; K6 –bijnik (powierzchnia górna-główka)-baryłka; K7 – baryłka-dźwignia rep.; K8 - dźwignia rep.-ogranicznik; K9,10 - dźwignia rep.-ograniczniki figury (górnny i dolny); K11 - młotek-struna; K12 - młotek-chwytnik; K13 - tłumik-struna; K14 – tłumik-ogranicznik; K15 - klawisz-łyżka tłumika; K16 - klawisz-ogranicznik (oparcie klawisza); K17 – klawisz-dno klawiatury;

3. Model matematyczny.

Podsumowując powyższe założenia równania ruchu wszystkich 6-ciu ciał zawierać będą w sobie składowe sił bezwładności, ciężkości i wzajemnego oddziaływanego na siebie poszczególnych elementów, oraz momenty tłumiące i pochodzące od sprężynek, w jakie zaopatrzona jest dźwignia i bijnik.

Istotnym uproszczeniem jest założenie, że właściwości sprężyste stykających się ciał, mimo, że wykazują one histerezę, można opisać funkcją dwóch zmiennych - chwilowej prędkości i głębokości penetracji jednego ciała względem drugiego bez odwoływanego się do całej historii zderzenia. Podobne uproszczenie wprowadziło do modelu tarcia, uzależniając je zarówno od nacisku jednego ciała na drugie, jak i od ich względnej prędkości w kierunku stycznym do powierzchni kontaktu bez potrzeby wykorzystywania prędkości z poprzedniego przedziału czasu (patrz p. 3.3.4). Klasyczny algorytm, który wymaga odwołania się do niej okazuje się tu niewydajny. Wprowadza on nieciągłości i czyni algorytm bardzo „wrażliwym” na małe oscylacje prędkości, co zaburza charakter całego zjawiska. Siłę tarcia uwzględniam tylko w trzech punktach kontaktu: pomiędzy pilotem i figurą, bijnikiem, dźwignią rep. i baryłką, oraz młotkiem i chwytnikiem. W pozostałych kontaktach wpływ tarcia jest znikomy.

Kolejnym istotnym uproszczeniem jest sprowadzenie geometrii kontaktów pomiędzy elementami do przenikania się półprostej z okręgiem. Zakładamy, że kontakt zachodzi *de facto* w punkcie, a nie na odcinku. Co więcej, ponieważ położenie tych punktów względem ciał zmienia się w bardzo małym stopniu (nie więcej, niż o 1mm), przypisuję je do ciał na stałe. Pokazuje to Rys.4. Wynikające stąd błędy w wartości momentów pochodzących od sił reakcji nie przekraczają przeważnie 1%.



Rys.4. Przypisanie na stałe punktów kontaktu.

Pozwala to w łatwy sposób opisać warunki kontaktu – głębokość penetracji, jej prędkość styczną i normalną do powierzchni oraz kierunek siły reakcji i tarcia. Takie rozwiązanie wydaje się optymalne w przypadku prostej geometrii dwuwymiarowej, gdzie punkty kontaktu łatwo można przewidzieć, a kontury elementów stanowią okręgi, bądź krzywe łamane. Naturalnie w przypadku geometrii bardziej skomplikowanej, zawierającej elementy o zarysie krzywoliniowym, gdzie miejsca zderzeń są trudne do określenia przed przeprowadzeniem symulacji znacznie wydajniejsze okazałyby się kontakty typu elipsa-elipsa. Takie rozwiązanie zawiera np. program Madymo stanowiący uniwersalne narzędzie do modelowania układów wieloczłonowych. Jednak w przypadku naszego układu kontakty typu okrąg – półprosta okazują się w pełni wystarczające.

Założyłem, że wszystkie ciała są sztywne, a odkształcenia pojawiają się jedynie w miejscach kontaktu. Ma to duże znaczenie przy modelowaniu kontaktu młoteczka ze struną i chwytnikiem. Ten ostatni osadzony jest na kilkucentymetrowym drucie, który odgina się sprężyste w momencie zatrzymania młoteczka. Zamiast modelować oddzielnie drut i filc chwytnika, potraktowałem je jako jedno ciało, co znacznie uprościło model.

3.3. Budowa algorytmu

Zanim przejdziemy do szczegółów algorytmu przyjrzymy się ogólnej konstrukcji programu. Pierwszym krokiem jest wczytanie danych takich, jak geometria, charakterystyki materiałów i parametry oraz zbudowanie na ich podstawie kompletnego zestawu informacji o układzie niezbędnych do działania programu. Następnie w pętli wykonywanej aż do zakończenia programu ma miejsce na przemian całkowanie układu równań i wizualizacja wyników na ekranie. Całkowanie odbywa się przy pomocy algorytmu *Fehlberg'a* V stopnia korzystającego z procedury *Row_Ruchu*. Wyznacza ona prawe strony równań postaci (12). W tym celu obliczana jest względna orientacja elementów, pomiędzy którymi może nastąpić kontakt, występujące pomiędzy nimi siły wzajemnych oddziaływań, w tym tarcie oraz momenty sprężynek i tłumienia. W trakcie wykonywania programu możliwe jest wprowadzanie drobnych poprawek geometrii w celu należytego wyregulowania mechanizmu.

Wielkością fizyczną stanowiącą w naszym układzie wymuszenie jest siła, z jaką palec naciska na klawisz. Jest ona na bieżąco wczytywana z pliku *Input.dat*. Zawiera on dane zebrane podczas eksperymentu mającego posłużyć późniejszej weryfikacji symulacji. Z kolei wielkości stanowiące wynik symulacji zapisywane są w pliku *Output.dat*. Są to położenia i prędkości klawisza i młoteczka zapisywane w zadanych odstępach czasu. Poniższe diagramy

3. Model matematyczny.

przedstawiają ogólną strukturę programu i procedury *Rów_Ruchu*. Sam program napisany jest w języku TURBO PASCAL for DOS i składa się z następujących plików:

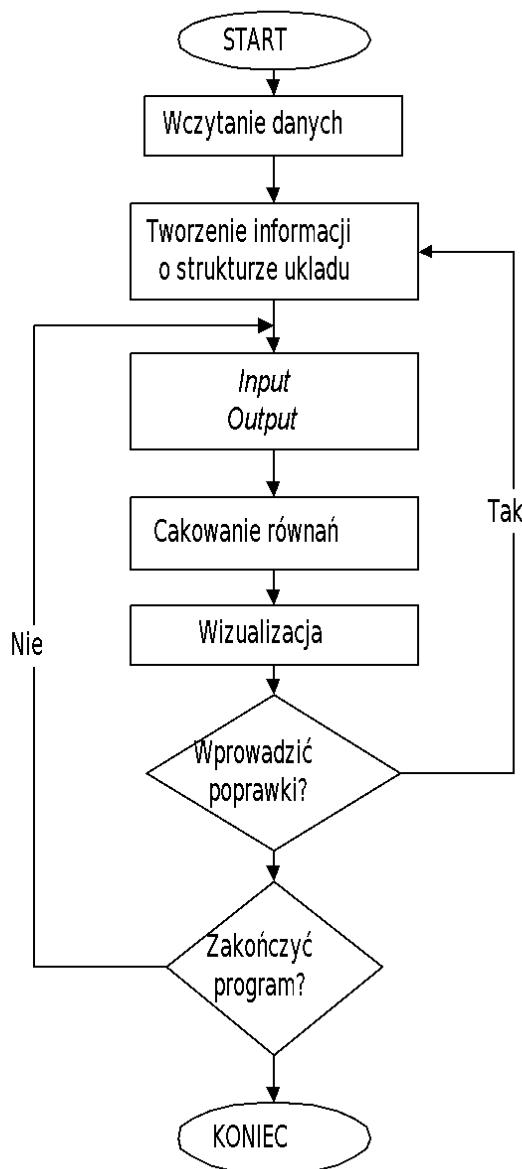
program główny – *Piano4.pas*

biblioteka deklaracji stałych i zmiennych programu - *Dane4.pas*

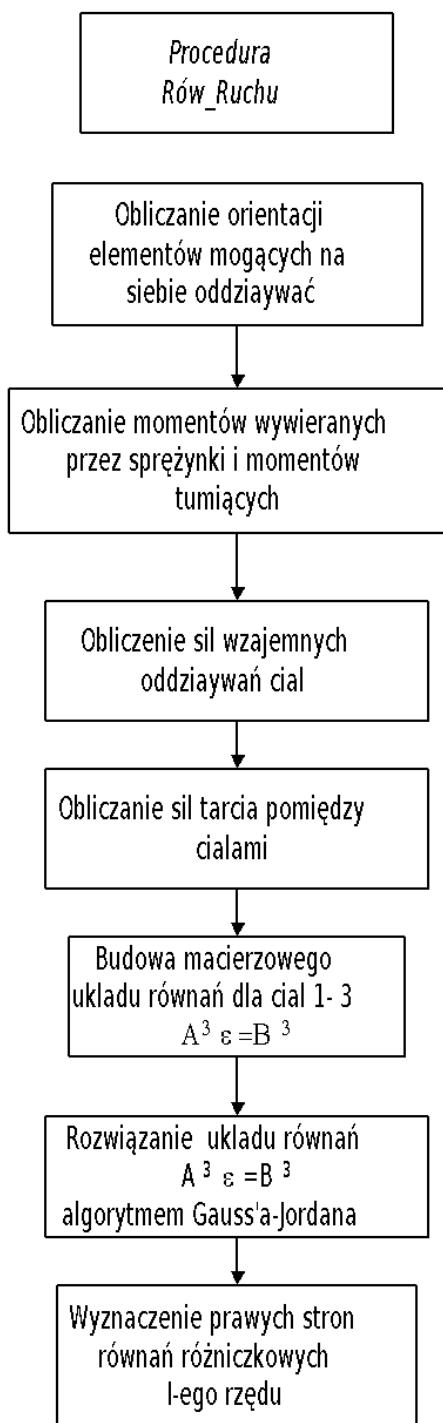
biblioteka procedur graficznych – *Grafika4.pas*

biblioteka procedur numerycznych – *Proc_Num4.pas*

pliki danych – *Geo4.dat, Kon4.dat, Map4.dat, Pop4.dat, Mat4.dat, Input4.dat*



Rys.5. Struktura programu.



Rys.6. Struktura procedury *Rów_Ruchu*.

Z teoretycznego punktu widzenia najważniejszym elementem algorytmu są równania ruchu, dlatego też od ich wyprowadzenia rozpoczęte omawianie całego modelu.

3.4. Równania ruchu

Ogólna najprostsza postać równań ruchu dla ciał nie wchodzących w skład żadnego łańcucha kinematycznego (w naszym przypadku dla młoteczka, tłumika i klawisza) ma postać:

$$\frac{d^2\alpha}{dt^2} = \frac{\Sigma M(\alpha, \omega)}{I} \quad (1)$$

gdzie:

$\vec{\omega}$ - wektor zawierający prędkości kątowe wszystkich ciał;

$\vec{\alpha}$ - wektor zawierający położenia kątowe wszystkich ciał;

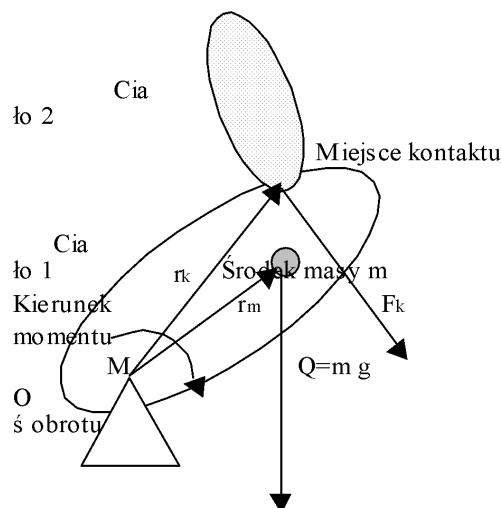
I – moment bezwładności liczony względem osi obrotu;

ΣM – suma momentów wszystkich sił działających na dane ciało będąca funkcją

wektorów $\vec{\omega}$ i $\vec{\alpha}$:

$$\Sigma M = M_{\text{gravitacji}} + M_{\text{sprężynek}} + M_{\text{tłumienia}} + M_{\text{kontaktów}}$$

Rys. 7. Pokazuje przykładowe siły i momenty działające na ciało.



Rys.7. Przykładowe siły działające na ciało 1; Q – siła grawitacji przyłożona w środku masy; r_m – wektor łączący oś obrotu ciała z środkiem masy; F_k – siła reakcji działająca w punkcie kontaktu z ciałem 2; r_k - wektor łączący oś obrotu ciała z umownym punktem kontaktu; M -suma momentu tłumiącego i momentu pochodzącego od sprężynek.

3. Model matematyczny.

Po zastąpieniu

$$\frac{d^2\alpha}{dt^2} \Rightarrow \varepsilon$$

, gdzie ε - przyspieszenie kątowe danego ciała;

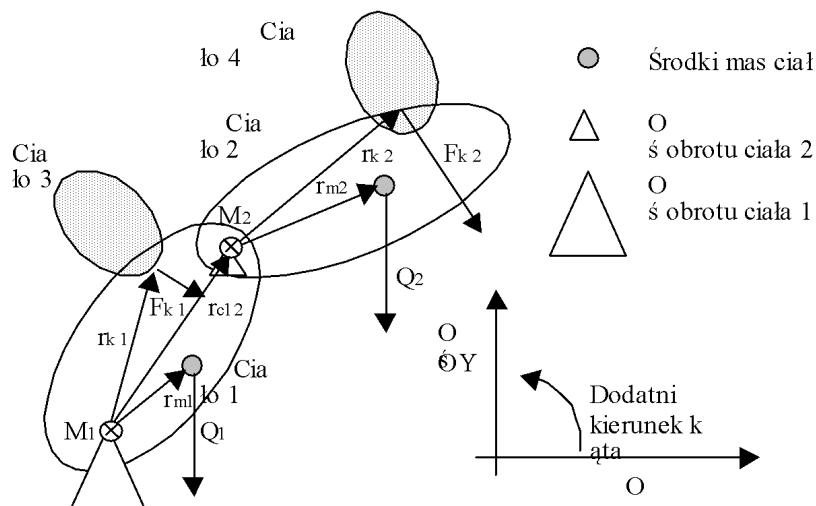
... i zastosowaniu zapisu wektorowego rów.(1) przybiera postać :

$$\begin{aligned} \varepsilon * I &= \Sigma M \\ \Sigma M &= \sum \overrightarrow{r_{k,i}} \times \overrightarrow{F_{k,i}} + Q * r_{m,x} + M_{tlumienia} + M_{sprezynek} \end{aligned} \quad (2)$$

, gdzie indeks i odpowiada kolejnym kontaktom w obrębie danego ciała;

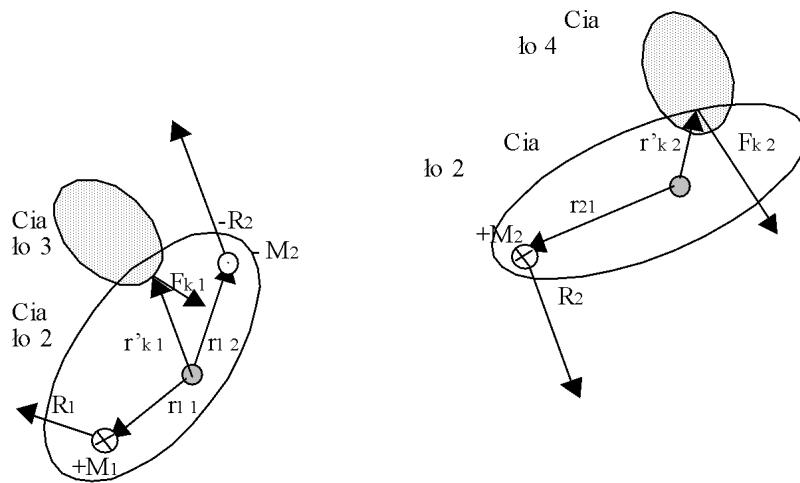
W przypadku łańcucha kinematycznego, jaki tworzą figura z bijnikiem i dźwignią, równania ruchu poszczególnych ciał przyjmują postać macierzową, jako, że pojawiają się pomiędzy nimi liczne sprzężenia. Aby wyznaczyć te równania należy „uwolnić od więzów” wszystkie ogniska łańcucha (Rys.8. i 9.) i napisać dla nich bilans momentów. Pojawiające się w nich reakcje w więzach obliczamy na bazie bilansu sił w każdym z ciał.

Aby wyprowadzić równania ruchu dla 3-elementowego łańcucha : figura-dźwignia-bijnik najprościej jest wyprowadzić je najpierw dla łańcucha 2-elementowego, a następnie przez analogię uzupełnić macierz równań o trzecie ciało. Aby właściwie uwzględnić wpływ reakcji w więzach, bilans momentów należy przeprowadzić nie względem osi obrotu, tak jak w przypadku rów.1., ale względem środka masy.



Rys.8. Przykładowe siły działające na łańcuch ciały 1 i 2 (konwencja oznaczeń ta sama, co na rys.5.; drugi indeks odpowiada numerowi ciała).

3. Model matematyczny.



Rys.9. Uwolnienie od więzów.

Bilans momentów działających na ciało 1 w łańcuchu 2-elementowym (Rys.6.):

$$\varepsilon_1 * I_{1,0} = \underline{\underline{r}}_{11} \times \underline{\underline{R}}_1 + \underline{\underline{r}}_{12} \times (-\underline{\underline{R}}_2) + \Sigma M_1; \quad (3)$$

$$\Sigma M_1 = \sum \underline{\underline{r}}_{k1,i} \times \underline{\underline{F}}_{k1,i} + r_{m1x} * Q_1 + \sum M_{tlum1,i} + \sum M_{sprez1,i}; \quad (3a)$$

, gdzie :

ε_1 , $I_{1,0}$ – przyspieszenie kątowe i moment bezwładności względem środka masy dla ciała 1;

$\underline{\underline{r}}_{k1,i}$ – wektory łączące środek masy ciała 1 z i-tymi kontaktami (rys.7.);

$\underline{\underline{r}}_{k2,i}$ – wektory łączące środek masy ciała 2 z i-tymi kontaktami;

$\underline{\underline{F}}_{k1,i}$ – siły reakcji i-tych kontaktów w ciele 1;

$\underline{\underline{F}}_{k2,i}$ – siły reakcji i-tych kontaktów w ciele 2;

$\overrightarrow{R}_1, \overrightarrow{R}_2$ – reakcje więzów :

$$R_{1y} = y''_1 * m_1 + R_{2y} - Q_1 - \sum F_{1ky,i}; \quad (4)$$

$$R_{1x} = x''_1 * m_1 + R_{2x} - \sum F_{1kx,i};$$

$$R_{2y} = y''_2 * m_2 - Q_2 - \sum K_{2ky,i};$$

$$R_{2x} = x''_2 * m_2 - \sum K_{2kx,i};$$

3. Model matematyczny.

Przyspieszenia punktów środków mas ciała 1 i 2 - $x''_1, y''_1, x''_2, y''_2$ wyznaczamy dwukrotnie różniczkując współrzędne środków mas m_1 i m_2 wyrażone w układzie kartezjańskim. Dla ułatwienia zapisu wprowadźmy następujące oznaczenia :

$$s_{11} = \sin(\alpha_{11}); \quad c_{11} = \cos(\alpha_{11});$$

, gdzie α_{11} jest kątem nachylenia wektora \underline{r}_{11} do osi OX.

Analogicznie oznaczamy $s_{12}, s_{21}, s_{c1}, s_{m1}, s_{m2}, c_{12}$, itd. Symbol danego wektora bez strzałki u góry oznacza jego moduł.

Współrzędne środków mas m_1 i m_2 wyrażone w układzie kartezjańskim:

$$x_1 = c_{m1} * r_{m1}; \quad y_1 = s_{m1} * r_{m1};$$

$$x_2 = c_{c12} * r_{c12} + c_{m2} * r_{m2}; \quad y_2 = s_{c12} * r_{c12} + s_{m2} * r_{m2};$$

Po dwukrotnym zróżniczkowaniu i uwzględnieniu zależności :

$$\dot{r}_{m1} = -\dot{r}_{11}; \quad \dot{r}_{m2} = -\dot{r}_{21},$$

$$\alpha'_{11} = \alpha'_{12} = \alpha'_{c1} = \omega_1; \quad \omega'_1 = \varepsilon'_1;$$

$$\alpha'_{21} = \omega_2; \quad \omega'_2 = \varepsilon_2;$$

otrzymujemy wartości przyspieszeń:

$$y''_1 = (s_{11} * \omega_1^2 - c_{11} * \varepsilon_1) * r_{11} \quad (5)$$

$$x''_1 = (c_{11} * \omega_1^2 + s_{11} * \varepsilon_1) * r_{11}$$

$$y''_1 = (-s_{c12} * \omega_1^2 + c_{c12} * \varepsilon_1) * r_{c12} + (s_{11} * \omega_1^2 - c_{11} * \varepsilon_1) * r_{11};$$

$$x''_1 = (-c_{c12} * \omega_1^2 - s_{c12} * \varepsilon_1) * r_{c12} + (c_{11} * \omega_1^2 + s_{11} * \varepsilon_1) * r_{11};$$

Jak wcześniej wspomniałem, na każde z ciał działa pewien moment tłumiący pochodzący od tarcia na osiach obrotu oraz oporów powietrza. Bijnik i dźwignia będące w łańcuchu odpowiednikiem ciała 2 zaopatrzone są w sprężynki. Ponieważ zaczepione są one jednym końcem do bijnika lub dźwigni, a drugim do figury – ciała 1, działają na nią tym samym momentem, co na ciało 2, tyle tylko, że skierowanym przeciwnie (patrz rys.7). Podobnie jest z momentem tłumiącym.

3. Model matematyczny.

Zatem rów. 3a. ma postać :

$$\begin{aligned} \Sigma M_1 = & + \sum (\vec{r}_{kl,i} \times F_{kl,i}) \\ & + r_{m1,x} * Q_1 \\ & + M_1 - M_2 \end{aligned} \quad (6)$$

, gdzie

$$M_1 = M_{\text{tum } 1} + M_{\text{spreż } 1};$$

$$M_2 = M_{\text{tum } 2} + M_{\text{spreż } 2};$$

Po wstawieniu (4) do (3) otrzymujemy:

$$\begin{aligned} \varepsilon_1 * I_{1,0} = & + r_{11x} * (y''_1 * m_1 + y''_2 * m_2 - Q_2 - \sum F_{k2y} - Q_1 - \sum F_{k1y}) \\ & - r_{11y} * (x''_1 * m_1 + x''_2 * m_2 - \sum F_{k2x} - \sum F_{k1x}) \\ & - r_{12x} * (y''_2 * m_2 - Q_2 - \sum F_{k2y}) \\ & + r_{12y} * (x''_2 * m_2 - \sum F_{k2x}) \\ & + \sum M_1; \end{aligned}$$

Uwzględniając (5) dostajemy :

$$\begin{aligned} \varepsilon_1 * I_{1,0} = & + r_{11x} * (s_{11} * \omega_1^2 - c_{11} * \varepsilon_1) * r_{11} * m_1 \\ & + r_{11x} * (-s_{c12} * \omega_1^2 + c_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{11x} * (s_{21} * \omega_2^2 - c_{21} * \varepsilon_2) * r_{21} * m_2 \\ & - r_{11x} * Q_2 - r_{11x} * \sum F_{k2y} - r_{11x} * Q_1 - r_{11x} * \sum F_{k1y} \\ & + r_{11y} * (-c_{11} * \omega_1^2 - s_{11} * \varepsilon_1) * r_{11} * m_1 \\ & + r_{11y} * (c_{c12} * \omega_1^2 + s_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{11y} * (-c_{21} * \omega_2^2 - s_{21} * \varepsilon_2) * r_{21} * m_2 \\ & - r_{11y} * \sum F_{k2x} - r_{11y} * \sum F_{k1x} \\ & + r_{12x} * (s_{c12} * \omega_1^2 - c_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{12x} * (-s_{21} * \omega_2^2 + c_{21} * \varepsilon_2) * r_{21} * m_2 \\ & + r_{12x} * Q_2 - r_{12x} * \sum F_{k2y} \\ & + r_{12y} * (-c_{c12} * \omega_1^2 - s_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{12y} * (c_{21} * \omega_2^2 + s_{21} * \varepsilon_2) * r_{21} * m_2 \\ & - r_{12y} * \sum F_{k2x} \\ & + \sum M_1; \end{aligned}$$

3. Model matematyczny.

Po uporządkowaniu i wprowadzeniu iloczynów skalarnych i wektorowych otrzymujemy :

$$\begin{aligned}
 & + \boldsymbol{\varepsilon}_1 * (I_{1,0} + (r_{11x}^2 + r_{11y}^2) * m_1 - \overrightarrow{r_{11}} \otimes \overrightarrow{r_{c12}} * m_2 + \overrightarrow{r_{12}} \otimes \overrightarrow{r_{c12}} * m_2) \\
 & + \boldsymbol{\varepsilon}_2 * (-\overrightarrow{r_{c12}} \otimes \overrightarrow{r_{21}} * m_2) = \\
 & = + \omega_1^2 * (-\overrightarrow{r_{11}} \times \overrightarrow{r_{11}} * m_1 + \overrightarrow{r_{11}} \times \overrightarrow{r_{c12}} * m_2 - \overrightarrow{r_{c12}} \times \overrightarrow{r_{c12}} * m_2) \\
 & + \omega_2^2 * (-\overrightarrow{r_{11}} \times \overrightarrow{r_{21}} * m_2 + \overrightarrow{r_{12}} \times \overrightarrow{r_{21}} * m_2) \\
 & - r_{11x} * Q_1 + (-r_{11x} + r_{12x}) * Q_2 - \overline{\Gamma_{11} \times \Sigma F_k} - \overline{\Gamma_{11} \times \Sigma F_{k2}} + \overline{\Gamma_{12} \times \Sigma F_{k2}}
 \end{aligned}$$

Korzystając z zależności :

$$-\overrightarrow{r_{11}} + \overrightarrow{r_{12}} = \overrightarrow{r_{c12}} ; \quad (7)$$

$$\overline{\Gamma_{m1}} = -\overline{\Gamma_{11}} ; \quad \overline{\Gamma_{m2}} = -\overline{\Gamma_{21}},$$

$$\overline{\Gamma}_{k1,i} - \overline{\Gamma_{11}} = \overline{\Gamma}_{k1,i}$$

, gdzie $\overline{\Gamma}_{k1,i}$ - wektory łączące oś obrotu ciała 1 z i-tymi kontaktami (rys.6.);

... twierdzenia Steiner'a :

$$I_{1,0} + r_{11}^2 * m_1 = I_1$$

, gdzie I_1 - moment bezwładności ciała 1 względem jego osi obrotu;

... i (6) otrzymujemy ostatecznie:

$$\begin{aligned}
 & + \boldsymbol{\varepsilon}_1 * (I_1 + \overrightarrow{r_{c12}} \otimes \overrightarrow{r_{c12}} * m_2) \\
 & + \boldsymbol{\varepsilon}_2 * (\overrightarrow{r_{c12}} \otimes \overrightarrow{r_{m2}} * m_2) = \\
 & = + \sum (\overline{\Gamma}_{k1,i} \times F_{k1,i}) + r_{m1,x} * Q_1 + M_1 \\
 & + \omega_2^2 * (\overrightarrow{r_{c12}} \times \overrightarrow{r_{m2}} * m_2) \\
 & + \overline{\Gamma_{c12} \times \Sigma F_{k2}} + r_{c12} * Q_2 - M_2 ;
 \end{aligned} \quad (8)$$

Pozostaje jeszcze wprowadzenie równań ruchu ciała 2:

3. Model matematyczny.

$$\varepsilon_2 * I_{2,0} = \underline{r}_{21} \times \underline{R}_2 + \Sigma M_2 ; \quad (9)$$

$$\Sigma M_2 = \sum \underline{r}_{k2,i} \times \underline{F}_{k2,i} + r_{m2,x} * Q_2 + \sum M_{tlum2,i} + \sum M_{sprez2,i} ; \quad (9a)$$

Wykonując te same operacje otrzymujemy :

$$\begin{aligned} \varepsilon_2 * I_{2,0} &= + r_{21x} * (y''_2 * m_2 - Q_2 - \Sigma F_{k2y}) \\ &- r_{21y} * (x''_2 * m_2 - \Sigma F_{k2x}) \\ &+ \Sigma M_2 ; \end{aligned}$$

Korzystając z (7) i zależności:

$$\underline{r}_{k2,i} - \underline{r}_{21} = \underline{r}_{k2,i}$$

, gdzie $\underline{r}_{k2,i}$ - wektory łączące oś obrotu ciała 2 z i-tymi kontaktami (rys.6.);

... twierdzenia Steiner'a :

$$I_{2,0} + r_{21}^2 * m_2 = I_2$$

, gdzie I_2 - moment bezwładności ciała 2 względem jego osi obrotu;

... otrzymujemy :

$$\begin{aligned} \varepsilon_2 * I_2 &= + r_{21x} * (-s_{c12} * \omega_1^2 + c_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{12x} * (s_{21} * \omega_2^2 - c_{21} * \varepsilon_2) * r_{21} * m_2 \\ &- r_{21x} * Q_2 - r_{21x} * \Sigma F_{k2y} \\ &+ r_{21y} * (-c_{c12} * \omega_1^2 - s_{c12} * \varepsilon_1) * r_{c12} * m_2 + r_{12y} * (c_{21} * \omega_2^2 + s_{21} * \varepsilon_2) * r_{21} * m_2 \\ &+ r_{21y} * \Sigma F_{k2x} \\ &+ \Sigma M_2 ; \end{aligned}$$

, gdzie

$$\begin{aligned} \Sigma M_2 &= + \Sigma (\underline{r}_{k2,i} \times \underline{F}_{k2,i}) \\ &+ r_{m2,x} * Q_2 \\ &+ M_2 \end{aligned}$$

i ostatecznie:

$$\begin{aligned}
 & + \varepsilon_1 * (\overrightarrow{r_{m2}} \otimes \overrightarrow{r_{c12}} * m_2) \\
 & + \varepsilon_2 * (I_2) = \\
 & = + \sum (\overline{\overline{r}_{k2,i} \times F_{k2,i}}) + r_{m2,x} * Q_2 + M_2 \\
 & + \omega^2 * (\overrightarrow{r_{m2}} \times \overrightarrow{r_{c12}} * m_2)
 \end{aligned} \tag{10}$$

Aby zbudować układ równań dla 3-elementowego łańcucha figura- bijnik- dźwignia rep. zamiast wyprowadzać skomplikowane równania dla trzech elementów porównałem równanie (2) oraz układ równań (8) i (10) zapisane w postaci macierzowej:

Równanie dla pojedynczego ciała :

$$\mathbf{A}^1 [\varepsilon_1] = \mathbf{b}^1$$

Układ równań dla 2-elementowego łańcucha :

$$\mathbf{A}^2 \cdot \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} = \mathbf{b}^2$$

Porównując macierze :

$$\mathbf{A}^1 = [I_1]$$

$$\mathbf{b}^1 = [\Sigma M_1]$$

$$\mathbf{A}^2 = \left[\begin{array}{c|c} I_1 + (\overrightarrow{r_{c12}} \otimes \overrightarrow{r_{c12}}) * m_2 & (\overrightarrow{r_{c12}} \otimes \overrightarrow{r_{m2}}) * m_2 \\ \hline \overrightarrow{r_{c12}} \otimes \overrightarrow{r_{m2}} * m_2 & I_2 \end{array} \right]$$

3. Model matematyczny.

$$\mathbf{b}^2 = \begin{bmatrix} \Sigma M_1 \\ + \omega_2^2 * (\vec{r}_{m2} \times \vec{r}_{c12}) * m_2 \\ + \vec{r}_{c12} \times \vec{\Sigma F}_{k2} + \vec{r}_{c12} * \vec{Q}_2 - \vec{M}_2 \\ \Sigma M_2 \\ + \omega_1^2 * (\vec{r}_{c12} \times \vec{r}_{m2}) * m_2 \end{bmatrix}$$

... łatwo zauważyc, które wyrazy pojawiły się wraz z rozszerzeniem równania (2) o ciało 2. Ponieważ w hierarchii łańcucha ciało 2 i 3 są na tym samym poziomie (Rys.10), aby uwzględnić w równaniach to ostatnie należy rozbudować macierz A^2 o jedną kolumnę i wiersz, a następnie uzupełnić właściwe pola kierując się wyżej wspomnianą analogią.

Układ równań dla 3-elementowego łańcucha :

$$\mathbf{A}^3 \cdot \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{bmatrix} = \mathbf{b}^3 \quad (11)$$

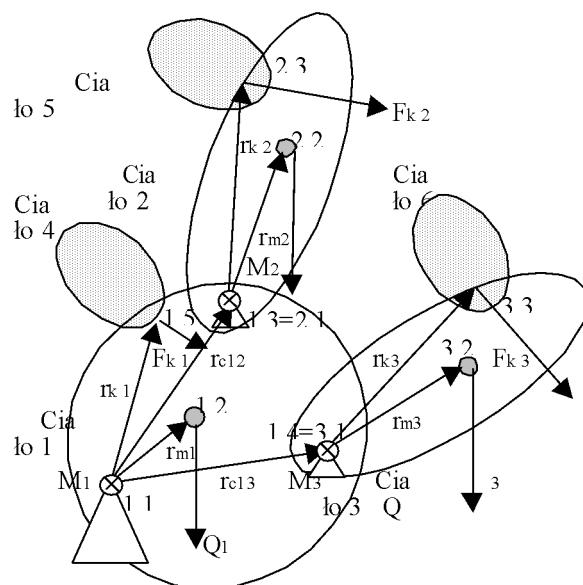
, gdzie

$$\mathbf{A}^3 = \begin{bmatrix} I_1 + (\vec{r}_{c12} \otimes \vec{r}_{c12}) * m_2 & (\vec{r}_{c12} \otimes \vec{r}_{m2}) * m_2 & (\vec{r}_{c13} \otimes \vec{r}_{m3}) * m_3 \\ \hline (\vec{r}_{c12} \otimes \vec{r}_{m2}) * m_2 & I_2 & 0 \\ \hline (\vec{r}_{c13} \otimes \vec{r}_{m3}) * m_3 & 0 & I_3 \end{bmatrix}$$

3. Model matematyczny.

$$\left[\begin{array}{l} \Sigma M_1 \\ + \omega_2^2 * (\vec{r}_{m2} \times \vec{r}_{c12}) * m_2 \\ + \vec{r}_{c12} \times \vec{\Sigma F_{k2}} + r_{c12} * Q_2 - M_2 \\ + \omega_3^2 * (\vec{r}_{m3} \times \vec{r}_{c13}) * m_3 \\ + \vec{r}_{c13} \times \vec{\Sigma F_{k3}} + r_{c13} * Q_3 - M_3 \\ \\ \Sigma M_2 \\ + \omega_1^2 * (\vec{r}_{c12} \times \vec{r}_{m2}) * m_2 \\ \\ \hline \hline \\ \Sigma M_3 \\ + \omega_1^2 * (\vec{r}_{c13} \times \vec{r}_{m3}) * m_3 \end{array} \right]$$

W przytoczonym poniżej algorytmie ciało 1 odpowiada figura, ciało 2 – dźwignia, a ciało 3- bijnik. Do rozwiązywania układu równań (11) wykorzystuję standardową procedurę Gaussa-Seidla otrzymując w wyniku wartości przyspieszeń wymienionych trzech ciał. Dla pozostałych ciał przyspieszenia obliczam bezpośrednio ze wzoru (2).



Rys.10. Przykładowy łańcuch 3-elementowy (Konwencja oznaczeń ta sama, co na rys.5. Przy punktach podano ich numery poprzedzone numerami ciał).

Kolejnym krokiem na drodze rozwiązania równań ruchu jest sprowadzenie równań różniczkowych II-go rzędu (1) do rzędu I-ego:

$$\begin{aligned} \frac{d\alpha}{dt} &= \omega \\ \frac{d^2\alpha}{dt^2} = \varepsilon &\Rightarrow \frac{d\omega}{dt} = \varepsilon \end{aligned} \quad (12)$$

Dopiero w tej postaci można je całkować przy pomocy jednej ze standardowych procedur numerycznych otrzymując w wyniku położenie i prędkość kątową każdego z ciał.

3.5. Obliczanie położenia i prędkości punktów we wsp. XY.

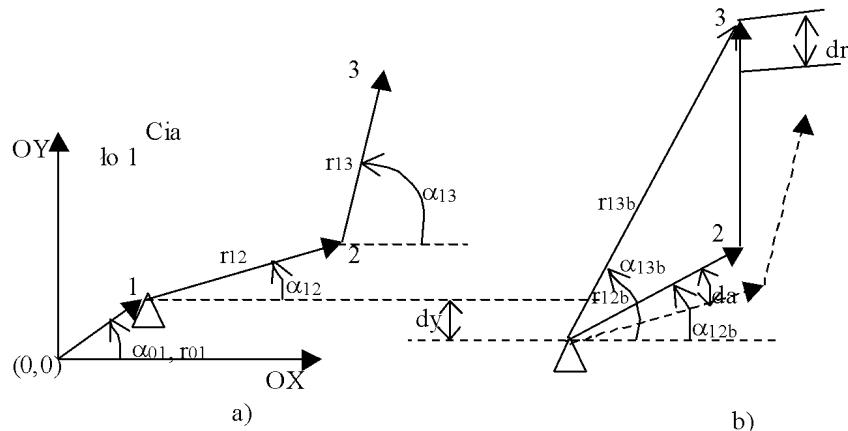
Jedną z głównych idei towarzyszących pisaniu programu było umożliwienie łatwego i szybkiego wprowadzania do geometrii układu drobnych poprawek już po skompilowaniu kodu. Jest to jeden z ważniejszych problemów pojawiających się przy tworzeniu układów w programie Madymo. W jego przypadku nawet niewielka zmiana orientacji któregoś z elementów wiąże się czasem z budowaniem całego fragmentu układu od początku. W przypadku mechaniki fortepianu wymagającej bardzo precyzyjnej regulacji, której nie sposób dokonać przed uruchomieniem symulacji wygoda wprowadzania poprawek geometrii jest sprawą kluczową radykalnie skracającą czas potrzebny na przygotowanie algorytmu do właściwej symulacji.

Cały proces budowania geometrii składa się z wielu etapów. Pierwszym z nich było sfotografowanie mechaniki i analiza zdjęcia przy pomocy programu TpsDIG ver.1.18. Pozwoliło to w łatwy sposób określić współrzędne w układzie kartezjańskim ok. 110 punktów (tzw. LandMark'ów - LM) tworzących kontury ciała – tzw. mapę. Po przeliczeniu ich z jednostek ekranowych – pixeli na mm zapisałem je w tablicy *Map* (tab.1). Następnie w tablicy *P* (tab.2) umieściłem informacje przypisujące punkty (LM) poszczególnym ciałom i porządkujące całą geometrię. Dla każdego punktu *p* należącego do ciała *c* określiłem punkt (*pp*) poprzedzający go w kolejności tworzenia konturu i jego ciało (*pc*), informację o tym, czy odcinek pomiędzy *p* i *pp* ma być rysowany (*r_*) i czy jego współrzędne i prędkość potrzebne są jedynie do wizualizacji, czy również do obliczenia sił wzajemnych oddziaływań ciał (tzw. kontaktów) (*k*), co wymaga częstszego ich obliczania. Kolejnym krokiem było

3. Model matematyczny.

stworzenie w tablicy *Pop* listy wszystkich poprawek możliwych do wprowadzenia w trakcie wykonywania programu.

Założymy, że w układzie z rys.11a chcemy nieznacznie obrócić odcinek 1-2, wydłużyć odc. 2-3 i obniżyć całe ciało 1. Zamiast w skomplikowany sposób przeliczać mapę (tab.1.) wystarczy wprowadzić odpowiednie poprawki (tab.3.).



Rys.11. Przykładowy układ dwóch odcinków.

Nr LM <i>i</i>	Wsp.x (<i>Map[ij].x</i>)	Wsp.y (<i>Map[ij].y</i>)
1.	0.0	0.0
2.	2.0	1.0
3.	3.0	3.0

Tab.1. Przykładowa mapa punktów.

c <i>i</i>	p <i>j</i>	k (<i>P[i,j].k</i>)	LM (<i>P[i,j].LM</i>)	pc (<i>P[i,j].pc</i>)	pp (<i>P[i,j].k</i>)	r (<i>P[i,j].r</i>)
0.	1.	0	1	0	0	0
1.	1.	1	1	0	1	0
	2.	0	2	1	1	1
	3.	1	3	1	2	1

Tab.2. Przykładowa tablica *P* porządkująca geometrię.

Nr poprawki <i>i</i>	Nazwa poprawki (<i>Pop[i].typ</i>)	Tablica danych	Rodzaj poprawki	Numer ciała (<i>Pop[i].c</i>)	Numer punktu	Wartość poprawki
-------------------------	---	----------------	-----------------	------------------------------------	--------------	------------------

3. Model matematyczny.

		$(Pop[i].ob)$	$(Pop[i].po)$		$(Pop[i].p)$	$(Pop[i].pop)$
1.	Obrót odc. 1-2	P	α	1	2	+0.01
2.	Wydłużenie odc. 2-3	P	r	1	3	+0.01
3	Obniżenie ciała 1	P	y	1	1	-0.01

Tab.3. Przykładowa lista poprawek.

Kolejność obliczania współrzędnych punktów jest następująca:

- w fazie inicjującej program (Procedura *Inic*):
 - obliczenie kierunków i długości wektorów łączących punkty ciała „zerowego” (tzw. Inertial Space) z początkiem układu współrzędnych $(0,0) - r_{01}, \alpha_{01} (P[0,j].r, P[0,j].a)$;
 - obliczenie kierunków i długości wektorów łączących ze sobą kolejne punkty ciał według porządku z tablicy $P - r_{12}, \alpha_{12}, r_{13}, \alpha_{13} (P[i,j].r, P[i,j].a)$;
 - ewentualna korekta wielkości $r_{12}, \alpha_{12}, r_{13}, \alpha_{13}$ o odpowiednie wartości z tablicy *Pop* - np. $\alpha_{12} = \alpha_{12} + d\alpha; r_{13} = r_{13} + dr$ (procedura *Popraw*);
 - obliczenie współrzędnych kartezjańskich punktów ($PO[i,j].x, PO[i,j].y$) jako sumy wektorów łączących kolejne punkty;
 - ewentualna korekta wielkości $PO[i,j].x$ i $PO[i,j].y$ o odpowiednie wartości z tablicy *Pop* - np. $PO[0,1].y = PO[0,1].y + dy$ (procedura *Popraw*);
 - kolejne obliczenie współrzędnych kartezjańskich punktów ($PO[i,j].x, PO[i,j].y$) jako sumy wektorów łączących kolejne punkty;
 - obliczenie kierunków i długości wektorów bazowych łączących kolejne punkty ciał z punktem bazowym (nr 1) ciała $- r_{12b}, \alpha_{12b}, r_{13b}, \alpha_{13b} (P[i,j].rb, P[i,j].ab)$;
- w pętli programu (Procedura *Obl_pkt*):
 - w procedurze *Row_Ruchu* (jeśli $P[i,j].k=1$) - obliczenie współrzędnych kartezjańskich wybranych punktów ($PO[i,j].x, PO[i,j].y$) na podstawie wektorów bazowych, których kierunek uzupełniony jest o aktualne położenie kątowe ($AL.[i]$) danego ciała.
 - podczas wizualizacji - obliczenie współrzędnych kartezjańskich wszystkich punktów;

3. Model matematyczny.

Oto przykładowy fragment kodu (procedura *Obl_pkt*) wykonujący ostatni etap obliczeń (*PO* jest tablicą zawierającą współrzędne x i y punktów):

```

for i:=1 to l_cial do
    for j:=1 to ST[i].l_p do
        begin
            if (P[i,j].k>=zak) then
                begin
                    pPO:=PO[P[i,1].pc,P[i,1].pp];
                    if(j=1) then
                        PO[i,j]:=pPO
                    else
                        begin
                            PO[i,j].x:=pPO.x+cos(P[i,j].ab+AL[i])*P[i,j].rb;
                            PO[i,j].y:=pPO.y+sin(P[i,j].ab+AL[i])*P[i,j].rb;
                        end;
                end;
        end;
    end;

```

(13)

, gdzie : i, j - numery odpowiednio ciał i punktów; l_cial – liczba ciał w układzie;

Poza geometrią każdemu z ciał przypisana jest w tablicy *ST* masa (*ST[i].Mas*), moment bezwładności względem osi obrotu (*ST[i].I*), ciężar (*Q*), nazwa (*ST[i].tx*) i kątowy zakres ruchów wykorzystywany do efektywnej wizualizacji wyników (*ST[i].zak*). Ponadto w proc *Inic* tworzone są automatycznie informacje na temat struktury ciała (podtablica *ST[i].S*, pola : $r, a, w(x,y), c, p$) zawierające długości, kierunki i składowe x i y wektorów budujących poszczególne stopnie łańcuch kinematycznego i ich punkty docelowe; ramiona sił kontaktów i przypisane im punkty (podtablica *ST[i].K*, pola: $r, a, w(x,y), c, p$); składowe sił kontaktów (podtablica *ST[i].KO*); współczynniki określające zwroty momentów tłumiących i sprężynek, o których mowa przy rów. (6) (podtablica *ST[i].M*); sumę momentów działających na ciało (*ST[i].MO*); liczbę punktów tworzących ciało (*ST[i].l_p*), liczbę kontaktów przypisanych ciała (*ST[i].l_k*) i liczbę stopni łańcucha (*ST[i].l_st*), gdzie 1-szy stopień stanowi zawsze dane ciało. Przykładowa tablica *ST* dla układu z rys.10 pokazana jest w tab.4.:

Nr ciała	Dane		
1.	$Mas=m_1; I=I_1; tx='ciało 1'; zak=30^\circ; Q=Q_1; MO=\sum M_1; l_p=4; l_k=1; l_st=1;$		
	Podtablica	Stopień 1	
	S =	$c=1; p=2; w=(r_{m1x}, r_{m1y}); a=a_{m1}; r=r_{m1};$	
	Podtablica	Kontakt 1	
	K=	$nK=1; c=1; p=5; a=a_{k1}; r=r_{k1};$	
	Podtablica	Kontakt 1	
	KO=	$x=F_{k1x}; y=F_{k1y};$	
	Podtablica	Ciało 1	Ciało 2
M=	+1	-1	-1

3. Model matematyczny.

2.	$Mas=m_2; I=I_2; tx='ciało 2'; zak=30^\circ; Q=Q_2; MO=\Sigma M_2; l_p=3; l_k=1; l_st=2;$			
	Podtablica	Stopień 1	Stopień 2	
	S =	$c=2; p=2; w=(r_{m2x}, r_{m2y}); a=a_{m2}; r=r_{m2};$	$c=1; p=3; w=(r_{c12x}, r_{c12y}); a=a_{c12}; r=r_{c12};$	
	Podtablica	Kontakt 1		
	K=	$nK=2; c=2; p=3; a=a_{k2}; r=r_{k2};$		
	Podtablica	Kontakt 1		
	KO=	$x=F_{k2x}; y=F_{k2y};$		
	Podtablica	Ciało 1	Ciało 2	Ciało 3
3.	M=	0	+1	0
	$Mas=m_3; I=I_3; tx='ciało 3'; zak=30^\circ; Q=Q_3; MO=\Sigma M_3; l_p=3; l_k=1; l_st=1;$			
	Podtablica	Stopień 1	Stopień 2	
	S =	$c=3; p=2; w=(r_{m3x}, r_{m3y}); a=a_{m3}; r=r_{m3};$	$c=1; p=4; w=(r_{c13x}, r_{c13y}); a=a_{c13}; r=r_{c13};$	
	Podtablica	Kontakt 1		
	K=	$nK=3; c=3; p=3; a=a_{k3}; r=r_{k3};$		
	Podtablica	Kontakt 1		
	KO=	$x=F_{k3x}; y=F_{k3y};$		
	Podtablica	Ciało 1	Ciało 2	Ciało 3
	M=	0	0	+1

Tab.4. Przykładowa tablica ST dla układu z rys.10. Kontakt nr 1 opisuje przenikanie ciała 1 i 4; nK=2 – ciała 2 i 5; nK=3 – ciała 3 i 6;

Dysponując powyższymi informacjami w łatwy sposób – różniczkując (13) można obliczyć prędkości liniowe wybranych punktów (zapisywane w tablicy VE). Oto odpowiedni fragment procedury *Obl_pkt*:

```

if (P[i,j].k=1) then
begin
    alf:=P[i,j].ab+AL[i];
    r:=P[i,j].rb;
    VE[i,j].w.x:=-sin(alf)*r*OM[i];
    VE[i,j].w.y:= cos(alf)*r*OM[i];
    for k:=2 to ST[i].l_st do
begin
    alf:=ST[i].S[k].a+AL[ST[i].S[k].c];
    r:=ST[i].S[k].r;
    cc:=ST[i].S[k].c;
    VE[i,j].w.x:=VE[i,j].w.x-sin(alf)*r*OM[cc];
    VE[i,j].w.y:=VE[i,j].w.y+cos(alf)*r*OM[cc];
end;
    VE[i,j].m:=dl_wek(p00,VE[i,j].w);
    VE[i,j].a:=k_wek(p00,VE[i,j].w);
end;

```

, gdzie: i i j - numery odpowiednio ciał i punktów;

$OM[i]$ – prędkość kątowa ciała;

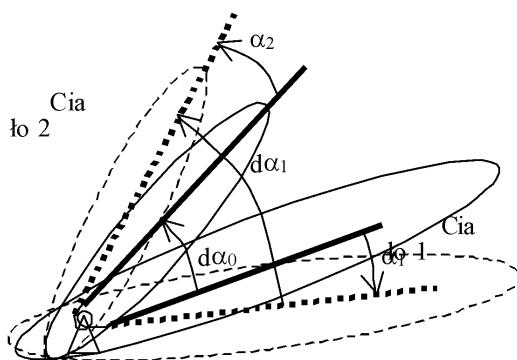
3. Model matematyczny.

$dl_wek()$, $k_wek()$ – funkcje obliczające długość i kierunek wektora;

$p00$ – wektor zerowy ($p00=[0,0]$);

3.6. Momenty sprężynek i tłumienia.

Zarówno bijnik, jak i dźwignia (odpowiednik ciała 2 na rys.12.) zaopatrzone są w sprężynki. Ponieważ są one umocowane w figurze (ciało 1 - bazowe), kąt odgięcia $d\alpha$ liczy się od kąta początkowego rozwarcia ciał $d\alpha_0$ równy każdej różnicy położenia kątowego obu ciał.



Rys.12. Działanie sprężynek (linią ciągłą narysowane są ciała w położeniu początkowym - $\alpha_1=\alpha_2=0$).

Z rys.12. wynika :

$$d\alpha = d\alpha_1 - d\alpha_0$$

$$d\alpha_1 = \alpha_2 - \alpha_1 + d\alpha_0$$

Zatem :

$$d\alpha = \alpha_2 - \alpha_1$$

Przyjmuję liniowy model sprężyny, której moment równy jest:

$$M_{sprz} = m_0 + m * d\alpha$$

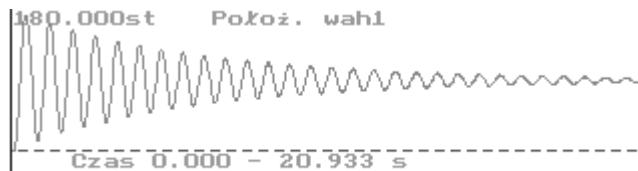
Tablica SP określa dla każdego z ciał ciało bazowe i współczynniki m_0 i m .

Zakładam, że tłumienie związane z tarciem na ośkach elementów zależy liniowo od ich prędkości kątowej :

$$M_{tłum} = -\omega * wsp_tłum;$$

3. Model matematyczny.

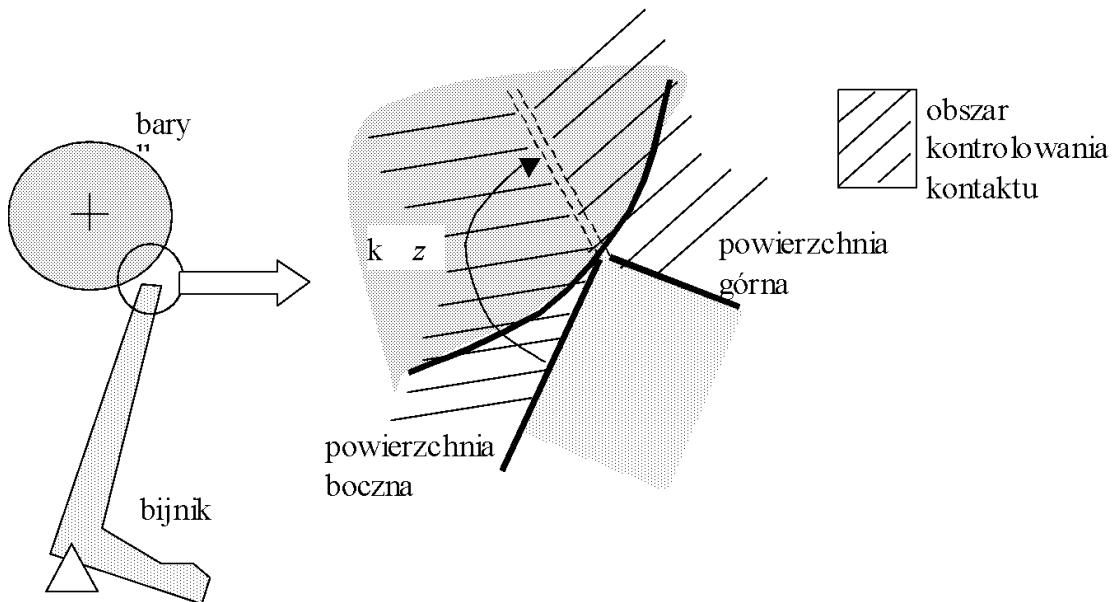
Wartość współczynnika tłumienia wyznaczyłem na podstawie prostego doświadczenia. Wyjąłem z modelu mechaniki jeden z elementów wraz z przymocowanym doń przegubem i wprowadziłem w ruch wahadłowy. Zmierzyłem początkowe odchylenie od pionu, czas, po jakim wahadło się zatrzymało i liczbę oscylacji. Porównując te dane z wynikami analogicznej symulacji komputerowej (wyk.1.) dobrąłem odpowiednie współczynniki dla wszystkich ciały. Zawierają się one w przedziale od ok.8e-5 do 1e-2.



Wyk.1. Przykładowa symulacja tłumienia oscylacji figury.

3.7. Sily wzajemnego oddziaływania ciał (naciski i tarcie).

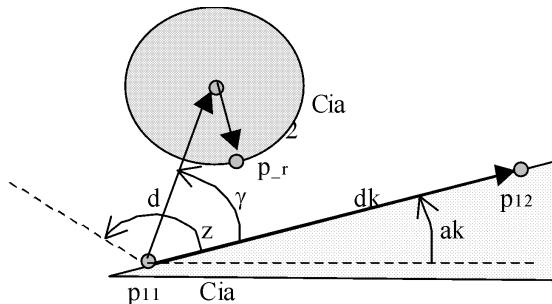
Jak już wcześniej wspomniałem, wszelkie kontakty ciał opisuję jako przenikanie się odcinka prostego z okręgiem. Dodatkowo dla każdego kontaktu określam tzw. kąt zakresu (z) definiujący obszar, w którym musi znajdować się środek okręgu, aby nastąpił "styk" powierzchni. Pozwala to opisać warunki kontaktu na złożonej geometrii, jak np. w przypadku kontaktu bijnika z baryłką. Może się on z nią zetknąć albo boczną, albo górną powierzchnią. Aby zamodelować taką geometrię używam *de facto* dwóch kontaktów - dla obu powierzchni tak określając zakres, aby oba obszary ze sobą graniczyły (Rys.13).



Rys.13. Przykład zamodelowania kontaktu bijnika z baryłką.

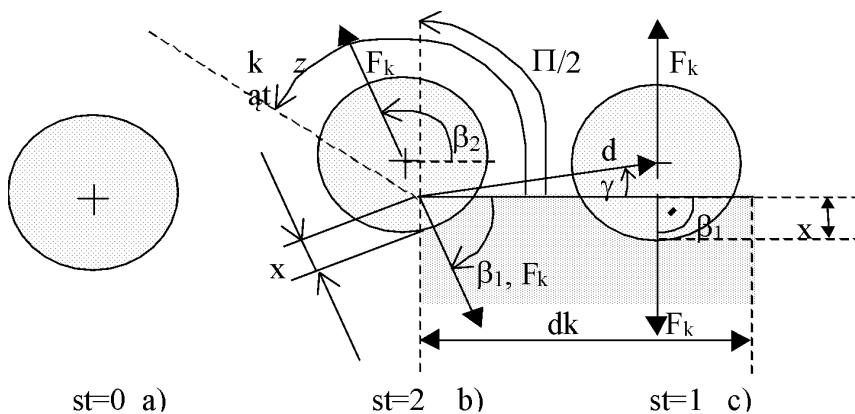
3. Model matematyczny.

Dane potrzebne do opisu kontaktów zawarłem w tablicy K . Są to numery tworzących odcinek i okrąg punktów oraz ich ciały ($p_{11}, p_{12}, p_2, p_r, c11$, itd.), kąt z , promień okręgu (rk), długość odcinka prostego i jego nachylenie (dk, ak), a także informacja, czy w dany kontakcie uwzględniane jest tarcie (tar). Sposób budowania kontaktu pokazuje rys.14.:



Rys.14. Sposób budowania kontaktu.

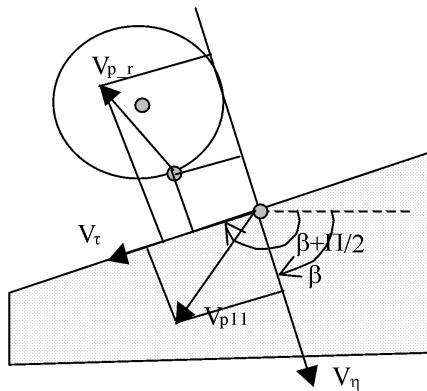
Wektor \vec{d} łączy punkt p_{11} z środkiem okręgu (Rys.15). Kąt γ liczony jest od wektora $\overrightarrow{p_{11}p_{12}}$ do wektora \vec{d} . Jeżeli $\gamma < 90^\circ$ - zachodzi przenikanie okręgu z obszarem „pod” odcinkiem (Rys.15.c)). Kierunek siły oddziaływania β jest wówczas do niego prostopadły. Jeżeli $z \geq \gamma \geq 90^\circ$, ma miejsce przenikanie punktu i okręgu (Rys.15.b)). Pozwala to na zamodelowanie kontaktu na narożu. Kiedy $\gamma > z$ lub $r > dk$ – kontakt nie następuje (Rys.15.c)). Zmienna st opisuje rodzaj kontaktu.



Rys.15. Rodzaje kontaktu (opisane zmienną st).

Aby obliczyć siłę reakcji i tarcie oprócz głębokości penetracji potrzebuję również składowe wzajemnej prędkości penetracji - normalną (V_n) i styczną (V_t) do powierzchni kontaktu (rys.16.). Prędkości te otrzymuję rzutując obliczone wcześniej prędkości punktów p_{11} i p_r na osie układu współrzędnych związanego z danym kontaktem. Oś normalna η pokrywa się co do kierunku z wektorem siły reakcji działającej na ciało 1, oś styczna τ obrócona jest względem η o kąt $-\Pi/2$.

3. Model matematyczny.



Rys.16. Składowe prędkości penetracji normalnej i stycznej dla okręgu i odcinka.

Poniższy algorytm (procedura *Sil_Kon*) wyznacza kierunek β siły, głębokość penetracji x i prędkość styczną V_t i normalną V_n . Posiadając te dane (zapisane w tablicy *KO*) procedura *R_mat* oblicza następnie wartość siły oddziaływań F_k . Oto uproszczony schemat procedury :

Dla $st=2$:

$$y=rk-d; \quad \beta_1 = \overrightarrow{p_2, p_{11}}; \quad \beta_2 = \overrightarrow{p_{11}, p_2}$$

Dla $st=1$:

$$y=rk-d * \sin(\beta); \quad \beta_1, \beta_2 = \pm ak;$$

```

for i:=1 to l_kon do
begin
  d:=dl_wek(PO[K[i].c2,K[i].p2],PO[K[i].c1,K[i].p11]);
  bet:=k_wek(PO[K[i].c2,K[i].p2],PO[K[i].c1,K[i].p11]);
  alf:=K[i].ak+AL[K[i].c1];
  gam:=nor_k(pi+bet-alf);
  x:=d;
  st_k:=0; {0-poza zakresem; 1- kon. z odc.; 2- kon. z pkt.}

  if(K[i].z>0) then
    if((0<gam) and (gam<K[i].z)) then
      if(abs(gam)<pip) then
        begin
          st_k:=1;
          bet:=nor_k(alf - pip);
        end
      else st_k:=2;
    if(K[i].z<0) then
      if((K[i].z<gam) and (gam<0)) then
        if(abs(gam)<pip) then
          begin
            st_k:=1;
            bet:=nor_k(alf + pip);
          end

```

3. Model matematyczny.

```

        else st_k:=2;
        if(K[i].z=0) then
            st_k:=2;

(...)

        if((st_k=0)or((x-K[i].rk>=0)and(st_k>0))) then {brak kontaktu}
        begin
            KO[i].N:=0;
            KO[i].x:=-abs(x-K[i].rk);
            KO[i].T :=0;
            KO[i].bT[1]:=0;
            KO[i].bT[2]:=0;
        end;
        if((x-K[i].rk<0)and(st_k>0)) then {kontakt}
        begin
            KO[i].x:=abs(x-K[i].rk);
            KO[i].N:=R_mat(K[i].mat,KO[i].x,KO[i].vw.n);
            KO[i].b[1]:=bet;
            KO[i].b[2]:=nor_k(bet+pi);
        end;

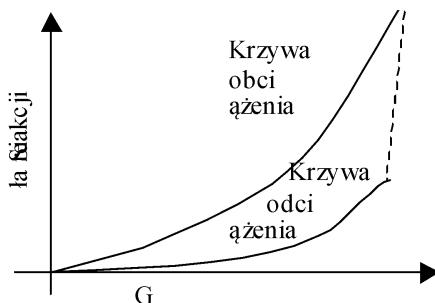
        KO[i].vw.w.x:=VE[K[i].c1,K[i].p11].x-VE[K[i].c2,K[i].p_r].x;
        KO[i].vw.w.y:=VE[K[i].c1,K[i].p11].y-VE[K[i].c2,K[i].p_r].y;
        KO[i].vw.m:=dl_wek(p00,KO[i].vw.w);
        KO[i].vw.a:=k_wek (p00,KO[i].vw.w);
        KO[i].vw.t:=cos(KO[i].vw.a-KO[i].b[1]+pip)*KO[i].vw.m;
        KO[i].vw.n:=cos(KO[i].vw.a-KO[i].b[1]+pi )*KO[i].vw.m;

(...)

end;

```

Materiały występujące w układzie nie są idealnie sprężyste. Pochłaniają znaczną część energii podczas zderzeń, a ich charakterystyki - siła od penetracji - wykazują właściwości histerezy (Wyk.2).



Wyk.2. Przykładowa pętla histerezy siły reakcji ciała.

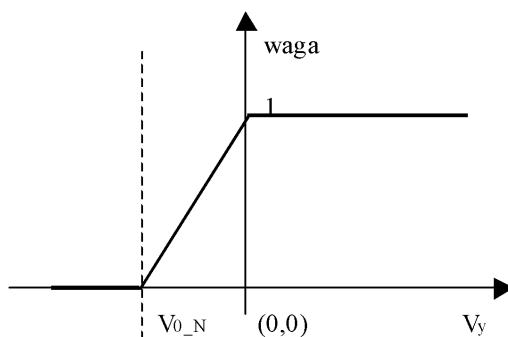
Aby w opisie matematycznym uwzględnić historię procesu nie odwołując się do wielkości z poprzednich przedziałów czasowych traktuję siłę reakcji jako funkcję dwóch zmiennych - penetracji i jej prędkości. Ma ona ogólną postać:

$$F = w F_{\text{obciążenia}} + (1-w) F_{\text{odciążenia}}$$

3. Model matematyczny.

, gdzie funkcja w pełni rolę wagi zależnej od prędkości i od parametru V_{0_N} ustalającego przedział prędkości, przy których następuje przejście z krzywej obciążenia na krzywą odciążenia lub odwrotnie (Wyk.3.):

$$w = \begin{cases} \text{dla } v \leq V_{0_N} & w = 0 \\ \text{dla } V_{0_N} < v \leq 0 & w = \frac{v - V_{0_N}}{V_{0_N}} \\ \text{dla } v > 0 & w = 1 \end{cases}$$



Wyk.3. Współczynnik wagowy.

Wartości sił F_{obc} i F_{odc} są interpolowane na podstawie serii danych uzyskanych doświadczalnie. Zapisane są one w tablicy *Mat* (podtablica *Mat[J].T*) wraz z innymi informacjami o występujących w układzie materiałach. Dotyczą one nie tyle pojedynczych materiałów, ile ich par (np. drewno w kontakcie ze skórą baryłki, metalowy pilot w kontakcie z filcem stopki, młoteczek w kontakcie ze struną, ogonek tłumika w kontakcie z chwytnikiem, itd.). Poza histerezą sprężystości określono dla nich również współczynniki tarcia statycznego i dynamicznego oraz odpowiednie parametry ($m_s, m_d, v0_N, v0_T, dF$).

Poniższy algorytm (funkcja *R_Mat*) oblicza funkcję wagową i interpoluje wartości sił reakcji:

```

if (v<Mat[m].v0_N) then
  w:=0;
  if ((v>=Mat[m].v0_N) and (v<0)) then
    w:=- (v-Mat[m].v0_N)/Mat[m].v0_N;
  if (v>=0) then
    w:=1;
  i:=0; ex:=false;
  repeat
    i:=i+1;
    if(i>l_p_mat) then
      begin
        i:=l_p_mat;

```

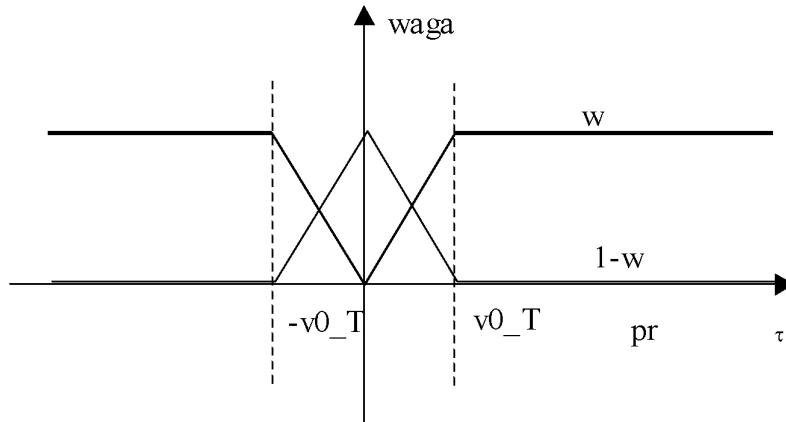
3. Model matematyczny.

```

x:=Mat[m].T[i].x
end;
if(x<=Mat[m].T[i].x) then
begin
  Fob:=(x-Mat[m].T[i-1].x)*Mat[m].T[i].aoB+Mat[m].T[i-1].yoB;
  Fod:=(x-Mat[m].T[i-1].x)*Mat[m].T[i].aoD+Mat[m].T[i-1].yoD;
  ex:=true;
end;
until (ex);
R_mat:=w*Fob+(1-w)*Fod;

```

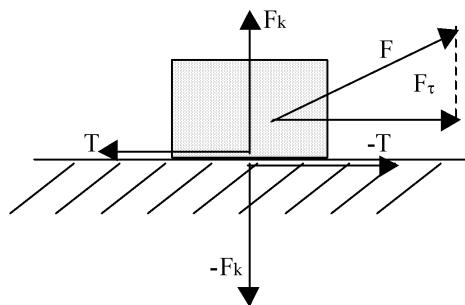
Podobny "chwyt" z funkcją wagową stosuję przy obliczaniu tarcia. Wyk.4. pokazuje jej przebieg. Pozwala ona na płynne przejście od tarcia statycznego do dynamicznego, które w odróżnieniu od klasycznych modeli tarcia nie wprowadza do algorytmu nieciągłości. Przytoczony w [11] standardowy algorytm porównuje bieżącą prędkość ciała z prędkością z poprzedniego przedziału czasu. W momencie, kiedy w wyniku tarcia następuje wyhamowanie i zmienia się zwrot prędkości, jest ona zerowana, a współczynnik tarcia przybiera wartość statyczną aż do czasu przekroczenia przez siłę wymuszającą wartości granicznej. Podobny schemat działania nie dość, że wymaga odwoływanego się do minionych wartości prędkości, to wprowadza liczne nieciągłości bardzo niewygodne dla procedur całkujących. Dlatego rozwiązanie z funkcją wagową wydaje się rozsądniejsze.



Wyk.4. Funkcja wagowa tarcia.

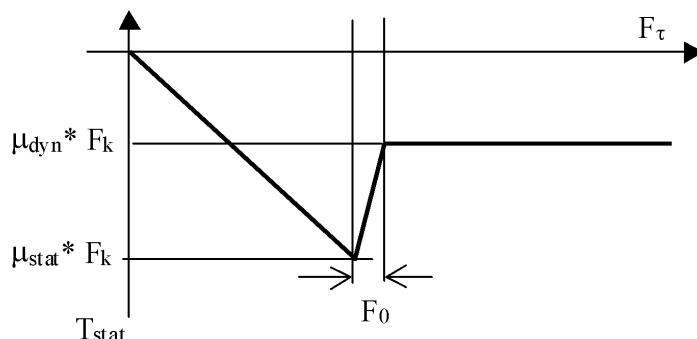
Aby przyjrzeć się mechanizmowi tarcia rozważmy na początek prosty układ – klocek leżący na płaskiej powierzchni – rys.17.:

3. Model matematyczny.



Rys.17. Układ klocek – podłoże.

W stanie spoczynku siła tarcia (statycznego) T , z jaką oddziałuje podłoże na klocek (zgodnie z III Zasadą Newtona klocek działa na podłoże z tą samą siłą T , ale z przeciwnym zwrotem) równoważy składową siły F wymuszającą ruch ciała styczną do powierzchni kontaktu $-F_\tau$. Jeżeli siła ta przekroczy wartość graniczną $\mu_{\text{stat}} * F_k$, tarcie zmaleje do wartości dynamicznej $\mu_{\text{dyn}} * F_k$ i rozpocznie się ruch ciała [12]. Zatem tarcie statyczne T_{stat} można opisać funkcją pokazaną na Wyk.5., a dynamiczne zależnością $T_{\text{dyn}} = \mu_{\text{dyn}} * F_k$.



Wyk.5. Funkcja tarcia statycznego.

Całkowite tarcie T wyniesie :

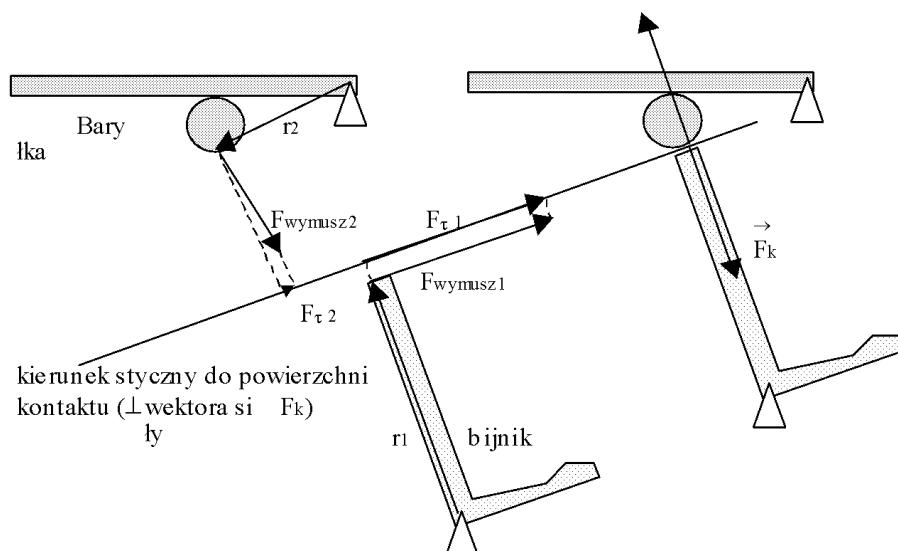
$$T = -w * \mu_{\text{dyn}} * F_k * \text{sign}(V_T) + (1-w) * T_{\text{stat}}(F_T) \quad (14)$$

Mechanizm tarcia pomiędzy elementami układu mechaniki różni się o tyle od przedstawionego powyżej, że zarówno „klocek”, jak i „podłoże” mogą się poruszać i na oba te ciała mogą działać siły wymuszające ruch. Wynika stąd, że prędkość styczna i normalna do powierzchni kontaktu będzie w rzeczywistości prędkością względną jednego ciała względem drugiego. Okazuje się, że w przypadku sił wymuszających jedno z ciał kontaktu można w naszym układzie - podobnie, jak na rys.17. - uznać za „stabilne podłoże”, na które w rzeczywistości nie działa żadne wymuszenie. Aby się o tym przekonać rozważmy przykładowy kontakt pomiędzy chwytnikiem i ogonkiem młoteczka (rys.1.), gdzie tarcie odgrywa zasadniczą rolę w prawidłowym działaniu mechanizmu. Pozwala ono na

3. Model matematyczny.

zakumulowanie energii sprężystej w sprężynie dźwigni repetycyjnej. W chwili, kiedy młoteczek jest zatrzymywany przez chwytnik uginając dźwignię, klawisz opiera się na filcu dna klawiatury równoważącym nacisk palca. W efekcie moment działający na zespół klawisza, a tym samym na chwytnik całkowicie się redukuje i co za tym idzie siła wymuszająca ze strony klawisza jest zerowa. W efekcie aby zamodelować tarcie statyczne w kontakcie chwytnik-młotek wystarczy rozważyć jedynie wymuszenie tego ostatniego.

W przypadku pozostałych kontaktów tarcie statyczne może się wprawdzie pojawić, kiedy oba ciała są w ruchu, a zatem na oba działają jakieś niezerowe momenty. Jednakże siły wymuszenia, jakie stąd wynikają rzutowane na kierunek styczny do powierzchni kontaktu w jednym z ciał zawsze są pomijalnie małe (rys.18).

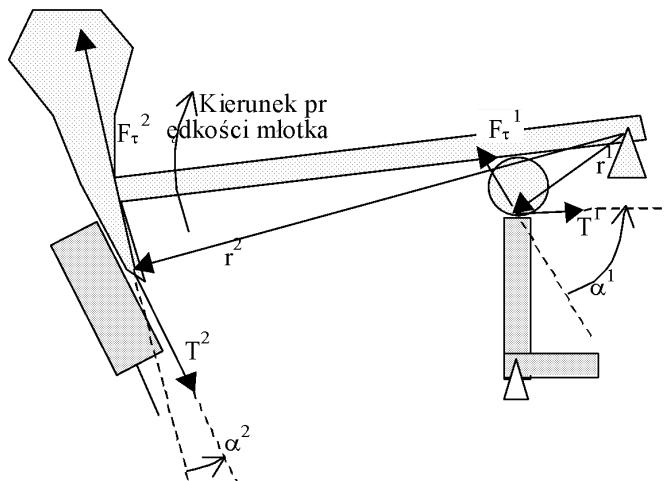


Rys.18. Porównanie udziału siły wymuszenia działającej na barłkę ($F_{\tau 2}$) i bijnik ($F_{\tau 1}$).

Analogiczna sytuacja będzie panować w kontakcie bijnik-dźwignia, gdzie brana jest pod uwagę jedynie siła działająca na dźwignię repetycyjną, oraz w kontakcie pilot-stopka - siła działająca na figurę.

Dzięki powyższemu uproszczeniu rozwiązany jest problem podziału momentu wymuszającego młotka na kilka kontaktów, w których pojawia się tarcie (kontakt baryłki z bijnikiem, dźwignią i chwytnikiem). Podział taki przypisujący każdemu z kontaktów inną wagę musiałby być proporcjonalny do udziału każdej z występujących sił tarcia statycznego w zrównoważeniu momentu wymuszającego. I tak np. tarcie pomiędzy dźwignią i baryłką posiadałoby mniejszą wagę, aniżeli między chwytnikiem i ogonkiem młoteczka z racji krótszego ramienia siły tarcia ($r^1 < r^2$) i mniej korzystnego kąta pomiędzy kierunkiem wektora siły wymuszenia i tarcia ($\cos(\alpha^1) < \cos(\alpha^2)$) (Rys.19).

3. Model matematyczny.



Rys.19. Rozkład sił wymuszających pomiędzy kontakty.

Poniżej pokazany jest przykładowy rozkład siły wymuszającej na dwa kontakty (F_τ^1 i F_τ^2) oraz współczynniki wagowe:

$$F_\tau^1 = \frac{m^1}{\sum m^i} * M / r^1 \cos(\alpha^1)$$

$$F_\tau^2 = \frac{m^2}{\sum m^i} * M / r^2 \cos(\alpha^2)$$

, gdzie :

$$m[1] = F_{k1} * \mu_{dynam} * \cos(\alpha_1)$$

$$m[2] = F_{k2} * \mu_{dynam} * \cos(\alpha_2)$$

Przyjmując jednak model tarcia „ciało- stabilne podłożę” podobne zabiegi stają się zbyteczne, jako że z wymienionych trzech kontaktów (kontakt baryłki z bijnikiem, dźwignią i chwytnikiem) jedynie ostatni „wykorzystuje” moment młoteczka. Informacja o tym, które ciało pełni rolę „stabilnego podłożą”, a które „ruchomego klocka” zawarta jest w tablicy $K[]/wym$. Poniżej przedstawiony jest algorytm liczący siłę tarcia w wybranych kontaktach (procedura *Sil_Tarcia*). Składowe styczne F_τ sił wymuszenia działające na każde ze stykających się ciał zapisywane są w tablicy $KO[nK].Ft[zw]$ (zw oznacza ciało opisane odcinkiem -1 lub okręgiem -2), po czym na bazie informacji z $K[]/wym$ wykorzystuje się tylko jedną z obliczonych sił ($KO[i].Ft[3]$). Następnie liczone są wagi ze wzoru (14) i ostatecznie siła tarcia i jej kierunek:

```
for i:=1 to l_cial do
    for j:=1 to ST[i].l_k do
        begin
```

3. Model matematyczny.

3.8. Rozwiązywanie równań ruchu dla zespołu figury.

Posiadając komplet informacji o siłach i momentach pojawiających się w układzie, można na ich podstawie wyprowadzić równania ruchu. Jak już wspomniałem w p.3.4. aby wyprowadzić równania dla zespołu figury należy rozwiązać układu równań (11).

Wykorzystana w tym celu procedura Gaussa-Seidla przekształca macierz **A** stopnia n z rów.(11) na sumę trzech macierzy [8], tj.:

$$A = L + D + U,$$

3. Model matematyczny.

, gdzie \mathbf{L} oznacza macierz trójkątną dolną, \mathbf{D} – macierz diagonalną, a \mathbf{U} – macierz trójkątną górną. Uwzględniając rozkład macierzy \mathbf{A} , układ równań (11) można zapisać w postaci następującej (wek. \mathbf{x} odpowiada $\boldsymbol{\varepsilon}$ z (11)):

$$(\mathbf{L}+\mathbf{D}+\mathbf{U})\mathbf{x}=\mathbf{b},$$

skąd

$$(\mathbf{L}+\mathbf{D})\mathbf{x} = -\mathbf{U}\mathbf{x} + \mathbf{b}.$$

Z powyższych zależności wynika następujący proces iteracyjny:

$$(\mathbf{L}+\mathbf{D})\mathbf{x}^{k+1} = -\mathbf{U}\mathbf{x}^k + \mathbf{b}.$$

tj.

$$\mathbf{x}^{k+1} = -(\mathbf{L}+\mathbf{D})^{-1} \mathbf{U}\mathbf{x}^k + (\mathbf{L}+\mathbf{D})^{-1}\mathbf{b}.$$

Jeśli promień spektralny ρ macierzy $-(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}$ jest mniejszy od 1, to powyższy proces iteracyjny jest zbieżny. Wynika z niego, że $(k+1)$ -sze przybliżenie i-tej składowej rozwiązania jest określone wzorem

$$x_i^{k+1} = \frac{\sum_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum_{j=i+1}^n a_{ij} x_j^k + b_i}{a_{ii}}, \quad i = 1, 2, \dots, n,$$

przy czym $a_{ii} \neq 0$.

Proces iteracyjny kończy się, gdy

$$\frac{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|}{\max(\|\mathbf{x}^{k+1}\|, \|\mathbf{x}^k\|)} \leq \varepsilon$$

przy $\mathbf{x}^{k+1} \neq 0$ lub $\mathbf{x}^k \neq 0$,

gdzie

$$\|\mathbf{x}\| = \max_{1 \leq i \leq n} |x_i|,$$

a ε oznacza dokładność, lub gdy $\mathbf{x}^{k+1} = 0$ i $\mathbf{x}^k = 0$, lub też, gdy liczba iteracji w procesie jest większa od maksymalnej.

Jak wynika z porównania wyników symulacji dla różnych ε (1e-5, 1e-6) powyższa metoda dla układu równań ruchu łańcucha kinematycznego wykazuje stabilność. Wykazanie jej na drodze teoretycznej jest bardzo kłopotliwe, bowiem o macierzy \mathbf{A} , poza tym, że jest

3. Model matematyczny.

symetryczna, trudno jest powiedzieć coś więcej – ustalić takie cechy, jak diagonalna dominacja, czy dodatnie określenie.

Poniżej pokazany jest fragment kodu tworzący macierz **A** i wektor **b** (proc. *Row_Ruchu*) oraz odwołanie do procedury *GaussSeidel*:

```
(...)
AA[1,1]:=ST[1].I+ST[2].Mas*il_sk(ST[2].S[2].w,ST[2].S[2].w)
          +ST[3].Mas*il_sk(ST[3].S[2].w,ST[3].S[2].w);
AA[1,2]:=           ST[2].Mas*il_sk(ST[2].S[2].w,ST[2].S[1].w);
AA[2,1]:=AA[1,2];
AA[2,2]:=ST[2].I;
AA[1,3]:=           ST[3].Mas*il_sk(ST[3].S[2].w,ST[3].S[1].w);
AA[3,1]:=AA[1,3];
AA[3,3]:=ST[3].I;

BB[1]:=+ST[1].MO
       +ST[2].Q*ST[2].S[2].w.x+il_wek(ST[2].S[2].w,ST[2].KO)
       +ST[3].Q*ST[3].S[2].w.x+il_wek(ST[3].S[2].w,ST[3].KO)
       +sqr(OM[2])*ST[2].Mas*il_wek(ST[2].S[1].w,ST[2].S[2].w)
       +sqr(OM[3])*ST[3].Mas*il_wek(ST[3].S[1].w,ST[3].S[2].w);

BB[2]:=+ST[2].MO
       +sqr(OM[1])*ST[2].Mas*il_wek(ST[2].S[1].w,ST[2].S[2].w);

BB[3]:=+ST[3].MO
       +sqr(OM[1])*ST[3].Mas*il_wek(ST[3].S[1].w,ST[3].S[2].w);

GaussSeidel (3,AA,BB,max_it_g,eps_g,EP,it_gau,st_gau);
```

Dla pozostałych ciał przyspieszenia obliczane są według (1):

```
for i:=n_mac+1 to l_cial do
  EP[i]:=ST[i].MO/ST[i].I;
```

Ostatnim krokiem jest wypełnienie wektora *f* (prawych stron równań (12)):

```
for i:=1 to l_cial do
begin
  f[i]:=      OM[i];
  f[i+l_cial]:=EP[i];
end;
```

3.9. Całkowanie równań ruchu.

Do całkowania równań ruchu wykorzystałem metodę Fehlberg'a. która w porównaniu z metodą Runge'go-Kutty 4-go rzędu okazała się zdecydowanie szybsza. Wybór metody jednokrokowej wynika ze specyfiki układu, a konkretniej z natury zjawiska zderzenia. Aby jego opis matematyczny był ciągły wprowadziłem opisane wyżej funkcje wagowe. Ich skuteczność w likwidowaniu nieciągłości zależy w równym stopniu od parametrów V_{0_N} i V_{0_T} i wielkości kroku całkowania. Musi być on odpowiednio mały, kiedy np. w którymś z kontaktów tarcie dynamiczne przechodzi w statyczne lub odwrotnie. Jednakże pomiędzy

3. Model matematyczny.

zderzeniami, kiedy funkcje równań ruchu znacznie się upraszczają, krok ten można zwiększyć (nawet o kilka rzędów) znacznie przyspieszając obliczenia. W metodach jednokrokowych możliwy jest dobór skoku w trakcie całkowania w celu jego optymalizacji. Cechą metod wielokrokowych, takich jak metoda Adams'a-Bashfort'a jest konieczność utrzymywania stałego kroku całkowania w ciągu całej symulacji. Rozwiązywanie takie mimo, że wymaga ono w obrębie jednego kroku rzadzkiego obliczania równań ruchu [7], byłoby mocno nieekonomiczne.

W przypadku prostych algorytmów liczących nieciągły skokowo ruch, jak np. zderzenia, zamiast w pełni symulować zjawisko odbicia stosuje się często zabieg „cofnięcia w czasie”. Ponieważ przy całkowaniu numerycznym nie zdarza się, aby chwila zetknięcia dwóch ciał następowała dokładnie na granicy kroku szacuje się czas t , jaki minął od tej chwili. Następnie całkując po ujemnym czasie t powraca się niemal dokładnie do momentu zderzenia ciał i odwraca się ich zwrot prędkości z uwzględnieniem ewentualnych strat [10]. W przypadku naszej symulacji takie uproszczenie oczywiście nie wchodzi w rachubę choćby z tego względu, że zależy nam na oszacowaniu sił reakcji w kontaktach, co w przypadku powyższej metody jest niemożliwe. Poza tym w naszym mechanizmie zderzenia te występują bardzo często i metoda „cofania się w czasie” zamiast przyspieszać obliczenia, mogłaby je w ogóle zatrzymać.

Metoda Fehlberga jako metoda jednokrokowa wykorzystuje wartości funkcji równań ruchu jedynie z ostatniego kroku całkowania. Należy ona do metod włożonych charakteryzujących się tym, że nakład obliczeń niezbędny w celu uzyskania wyniku za pomocą metody rzędu $K+1$ wytarcza w celu uzyskania wyników za pomocą metody rzędu K , bez dokonywania dodatkowych obliczeń [1]. Jest to szczególnie istotne przy oszacowaniu błędu. Istotą metody Fehlberga jest tworzenie pary metod włożonych K -tego rzędu i M -etapowej oraz $(K+1)$ -ego rzędu i $(M+1)$ -etapowej. Metoda Fehlberga należy do klasy metod Rungego-Kutty, w której pierwszym etapem jest obliczenie wektora \mathbf{K}_j (indeks $j=1,2,\dots,M+1$ odpowiada kolejnym numerom etapów metody) :

$$\mathbf{K}_j = h_j \mathbf{F} \left[\mathbf{Y}_i + \sum \beta_{jl} \mathbf{K}_l, t_i + \alpha_j h_i \right] \quad (15)$$

gdzie:

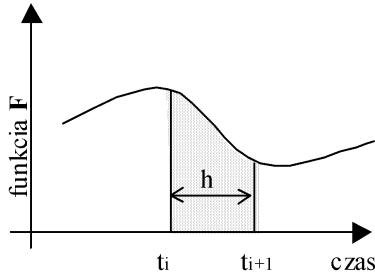
$h_i = t_{i+1} - t_i$ - krok całkowania równy szerokości całkowanego przedziału czasowego
(indeks i odpowiada numerowi przedziału - wyk.6.).

3. Model matematyczny.

α_j, β_{jl} – odpowiednie współczynniki metody ($\alpha_1=0; \beta_{1l}=0;$)

F – funkcja obliczająca prawe strony rów.(12), której argumentami jest czas t_i otwierający przedział i wyniki obliczeń z początku przedziału (dla t_i).

\mathbf{Y} – wektor wielkości różniczkowanych w (12) – położenie i prędkość ciał.



Wyk.6. Przedział całkowania.

Oznaczmy przez \mathbf{Y}_{i+1} wyniki obliczeń za pomocą metody (M+1) etapowej

$$\mathbf{Y}_{i+1} = \mathbf{Y}_i + \sum_{j=1}^{M+1} w_j \mathbf{K}_j \quad (16)$$

natomiast przez \mathbf{Y}^*_{i+1} wyniki obliczeń za pomocą metody (M) etapowej

$$\mathbf{Y}^*_{i+1} = \mathbf{Y}_i + \sum_{j=1}^{M+1} w_j^* \mathbf{K}_j \quad (17)$$

przy czym wektory \mathbf{K}_j określone wzorem (15) są wspólne dla wzorów (16) i (17).

Niech $\mathbf{Y}(t_i)$ oraz $\mathbf{Y}(t_i+h)$ będą dla ustalonego t_i oraz t_i+h wartościami rozwiązania dokładnego układu równań ruchu (12). Oznaczamy przez $\underline{\mathbf{Y}}(t_i+h;h)$ wyniki obliczeń metody (M+1) etapowej rzędu K+1 natomiast przez $\mathbf{Y}^*(t_i+h;h)$ metody M- etapowej rzędu K na podstawie wartości dokładnej $\mathbf{Y}(t_i+h)$

$$\underline{\mathbf{Y}}(t_i+h;h) = \mathbf{Y}(t_i) + \sum_{j=1}^{M+1} w_j \mathbf{K}_j \quad (18)$$

$$\mathbf{Y}^*(t_i+h;h) = \mathbf{Y}(t_i) + \sum_{j=1}^M w_j^* \mathbf{K}_j \quad (19)$$

Zgodnie z określeniem błędu aproksymacji w pierwszym przybliżeniu dla metod rzędu K+1 oraz K otrzymuje się odpowiednio :

$$\mathbf{E}(t_i+h;h) = -\underline{\mathbf{Y}}(t_i+h;h) + \mathbf{Y}(t_i+h) \approx \mathbf{E}'_{K+1}(t_i)h^{K+2} \quad (20)$$

$$\mathbf{E}^*(t_i+h;h) = -\mathbf{Y}^*(t_i+h;h) + \mathbf{Y}(t_i+h) \approx \mathbf{E}^{*_{K+1}}(t_i)h^{K+2} \quad (21)$$

Odejmując równania (20) i (21) stronami oraz uwzględniając wzory (18) i (19) błąd aproksymacji $\mathbf{E}^*(t_i+h;h)$ wyraża się w pierwszym przybliżeniu następująco

$$\mathbf{E}^*(t_i+h;h) \cong \sum_{j=1}^{M+1} (w_j - w_j^*) \mathbf{K}_j \quad (22)$$

gdzie: $w_{M+1}^* = 0$.

Zgodnie więc z metodą Fehlberga wykonujemy taki krok całkowania h aby norma wektora błędu $\mathbf{E}^*(t_i+h;h)$ określona wzorem (22) była mniejsza od z góry zadanego ε :

$$\|\mathbf{E}^*(t_i+h;h)\| < \varepsilon \quad (23)$$

Wzór (21) oszacowuje błąd aproksymacji metody M etapowej rzędu K. Jeżeli będzie zachodził warunek (23) dla danego kroku h to wykonamy obliczenie \mathbf{X}_{i+1} zgodnie ze wzorem (15), tj. metodą $(M+1)$ etapową rzędu $K+1$. Postępowanie takie czyni oszacowanie błędu bardziej wiarygodnym bo zaakceptowana została metoda dla rzędu niższego K. Należy przy tym zauważyć, że oszacowanie to nie wymaga dodatkowych obliczeń wartości funkcji $\mathbf{F}(\mathbf{X}, t)$ prawej strony równań (12) tak, jak ma to miejsce w podstawowej metodzie Rungego-Kutty.

Wyznaczenie wartości współczynników $\alpha_j(a)$ i $\beta_{jl}(b)$ oraz liczb $w_j^*(wE)$ i $w_j(wY)$ we wzorze (22) wymaga bardzo pracochłonnych wyprowadzeń, które w niniejszej pracy pomijam. Poniżej zamieszczam gotowy algorytm dla pary metod włożonych rzędu 4 i 5 i odpowiadający mu fragment kodu (procedura *krok*):

$$\begin{aligned} \mathbf{K}_1 &= h\mathbf{F}(\mathbf{Y}_i, t_i) \\ \mathbf{K}_2 &= h\mathbf{F}\left(\mathbf{Y}_i + \frac{1}{4}\mathbf{K}_1, t_i + \frac{1}{4}h\right) \\ \mathbf{K}_3 &= h\mathbf{F}\left(\mathbf{Y}_i + \frac{3}{32}\mathbf{K}_1 + \frac{9}{32}\mathbf{K}_2, t_i + \frac{3}{8}h\right) \end{aligned}$$

3. Model matematyczny.

$$\begin{aligned}\mathbf{K}_4 &= hF\left(\mathbf{Y}_i + \frac{1932}{2197} \mathbf{K}_1 - \frac{7200}{2197} \mathbf{K}_2 + \frac{7296}{2197} \mathbf{K}_3, t_i + \frac{12}{13} h\right) \\ \mathbf{K}_5 &= hF\left(\mathbf{Y}_i + \frac{439}{216} \mathbf{K}_1 - 8 \mathbf{K}_2 + \frac{3680}{513} \mathbf{K}_3 - \frac{845}{4104} \mathbf{K}_4, t_i + h\right) \\ \mathbf{K}_6 &= hF\left(\mathbf{Y}_i - \frac{8}{27} \mathbf{K}_1 + 2 \mathbf{K}_2 - \frac{3544}{2565} \mathbf{K}_3 + \frac{1859}{4104} \mathbf{K}_4 - \frac{11}{40} \mathbf{K}_5, t_i + \frac{1}{2} h\right) \\ \mathbf{E} &= \left(\frac{16}{135} - \frac{25}{216} \right) \mathbf{K}_1 + \left(\frac{6656}{12825} - \frac{1408}{2565} \right) \mathbf{K}_3 + \left(\frac{28561}{56430} - \frac{2197}{4104} \right) \mathbf{K}_4 + \left(-\frac{9}{50} + \frac{1}{5} \right) \mathbf{K}_5 + \frac{2}{55} \mathbf{K}_6 \\ \mathbf{Y}_{i+1} &= \mathbf{Y}_i + \frac{16}{135} \mathbf{K}_1 + \frac{6656}{12825} \mathbf{K}_3 + \frac{28561}{56430} \mathbf{K}_4 - \frac{9}{50} \mathbf{K}_5 + \frac{2}{55} \mathbf{K}_6\end{aligned}$$

```

a : array[1..6] of real=(0,1/4,3/8,12/13,1,1/2);
b : array[1..6,1..5] of real=
((0,          0,          0,          0,          0),
 (1/4,         0,          0,          0,          0),
 (3/32,        9/32,       0,          0,          0),
 (1932/2197, -7200/2197, 7296/2197, 0,          0),
 (439/216,    -8,         3680/513, -845/4104, 0),
 (-8/27,       2,         -3544/2565, 1859/4104, -11/40));
wE : array[1..6] of real=(1/360,0,-128/4275,-2197/75240,1/50,2/55);
wY : array[1..6] of real=(16/135,0,6656/12825,28561/56437,-9/50,2/55);

fun(x0,n,y0,ff);
  for i:=1 to n do K[1,i]:=ff[i]*hh;
  for l:=2 to 6 do
    begin
      for i:=1 to n do
        begin
          yy[i]:=y0[i];
          for j:=1 to l-1 do
            yy[i]:=yy[i]+b[l,j]*K[l-1,i];
        end;
      fun(x0+a[l]*hh,n,yy,ff);
      for i:=1 to n do K[l,i]:=ff[i]*hh;
    end;

    for i:=1 to n do
      begin
        y1[i]:=y0[i];
        E[i]:=0;
        for j:=1 to 6 do
          begin
            y1[i]:=y1[i]+wY[j]*K[j,i];
            E[i]:=E[i]+wE[j]*K[j,i];
          end;
      end;
  end;

```

Opracowaną procedurę *krok* można wykorzystać do wykonania kolejnych kroków obliczeniowych i do określenia długości tych kroków.

Niech $\mathbf{Y}(t_i)$ będzie dla ustalonego t_i wartością rozwiązania dokładnego układu (12). Oznaczmy przez $\underline{\mathbf{Y}}(t_i+H; H)$, wynik obliczeń metody $(M+1)$ -etapowej rzędu $K+1$ na

3. Model matematyczny.

podstawie $\mathbf{Y}(t_i)$ (wzór (18)) w jednym kroku obliczeń o długości H. Zgodnie z określeniem błędu aproksymacji w pierwszym przybliżeniu metody rzędu K (wzór (21)) oraz na mocy wzoru (22) otrzymuje się :

$$\mathbf{E}^*(t_i+H;H) = \mathbf{E}^*(t_i)H^{K+1} \cong \sum_{j=1}^{M+1} (w_j - w_j^*) \mathbf{K}_j \quad (24)$$

gdzie: \mathbf{K}_j określa się wzorem (15).

Wstępnie ustalona wartość kroku H pozwala na oszacowanie błędu aproksymacji w odniesieniu do H^{K+1} (wzór (15)). W każdym kroku całkowania można postawić następujący problem: dla danych $\mathbf{Y}(t_i)$, t_i określić możliwie dużą wartość kroku h tak, by norma błędu dyskretyzacji:

$$\mathbf{E}^*(t_i+h;h) = \mathbf{E}^*(t_i)h^{K+1} \quad (25)$$

po wykonaniu jednego kroku o tej długości pozostawała nadal mniejsza od pewnego ε . W praktyce dokładność obliczeń ε kreśla się dwoma parametrami ε_w – tolerancja błędu względnego oraz ε_a – tolerancja błędu bezwzględnego, tj.:

$$\varepsilon = \varepsilon_w \max \left\{ \|\mathbf{Y}(t)\| : t \in [t_i, t_i + h] \right\} + \varepsilon_a \quad (26)$$

gdzie:

$$\max \left\{ \|\mathbf{Y}(t)\| : t \in [t_i, t_i + h] \right\} \cong \|\tilde{\mathbf{Y}}_i\| \quad (27)$$

$$\tilde{y}_{ij} = \frac{1}{2} (|y_j(t_i)| + |y_j(t_i + H)|) \quad - \quad \text{współrzędne wektora } \tilde{\mathbf{Y}}_i \quad (28)$$

$$\|\tilde{\mathbf{Y}}_i\| = \max |\tilde{y}_{ij}| \quad - \quad \text{norma wektora} \quad (29)$$

Krok całkowania h dobiera się więc tak, aby norma wektora błędu (24) była równa ε (26), tj.:

$$\|\mathbf{E}^*(t_i + h, h)\| = \|\tilde{\mathbf{Y}}_i\| \varepsilon_w + \varepsilon_a \quad (30)$$

Wyznaczając z równania (24) $\mathbf{E}^*(t_i)$ i podstawiając do równania (25) otrzymuje się zgodnie z (30) :

3. Model matematyczny.

$$\frac{1}{\alpha} = \frac{H}{h} = \sqrt[|K+1|]{\frac{\left\| \sum_{j=1}^{M+1} (w_j - w_j^*) \mathbf{K}_j \right\|}{\|\tilde{\mathbf{Y}}\|_{\varepsilon_w} + \varepsilon_a}} \quad (31)$$

Wzór (31) daje praktyczny sposób doboru kroku całkowania h . Mając na uwadze, że wyprowadzenia powyższe mają charakter szacunkowy zamiast $h=\alpha H$ przyjmuje się jako bezpieczniejsze $h=0,9\alpha H$. Aby uniknąć nadmiernego wzrostu długości kroku całkowania przyjmuje się ograniczenie $\alpha \leq 5$. Procedura doboru skoku zostaje przerwana, jeżeli $h < h_{\min}$ lub liczba pętli procedury $it > it_max$, co sygnalizowane jest dźwiękowo :

```

for i:=1 to n do y0[i]:=y[i];
  if(x1>x0) then
    repeat
      it:=0;
    repeat
      stan:=-1;
      krok(x0,h,y0,y1,E,n,fun);
      dop_b1:=eps*max(n,y1)+eta;
      bl_:=max(n,E);
      alf:=exp(-1/6*ln(bl_/dop_b1))*0.9;
      if (alf>5) then alf:=5;
      h:=alf*h;
      it:=it+1;
      if (bl_>dop_b1) then stan:=-1
      else stan:=0;
      if (h<mh) then stan:=1;
      if (it>max_it) then stan:=2;
      if(stan>0) then
        sound(1000)
      else nosound;
      until (stan>=0);
      x0:=x0+h;
      for i:=1 to n do y0[i]:=y1[i];
    until ((x0>=x1)or(stan>0));
    for i:=1 to n do y[i]:=y1[i];
  
```

Pozostaje jeszcze kwestia doboru parametrów ε_w i ε_a . Parametr ε_w nie może być mniejszy od podwojonej dokładności maszynowej, która dla używanych w symulacji zmiennych typu *double* wynosi 1e-16. Najprościej jest wykonać kilka serii obliczeń dla różnych tolerancji i na podstawie różnic w wynikach dobrać optymalne ε_w i ε_a .

3.10. Regulacja mechanizmu.

Proces regulacji mechaniki fortepianu jest niezmiernie złożony i pracochłonny. Przeciętnie zabiera on wprawnemu stroicielowi od kilku do kilkunastu godzin. Naturalnie w przypadku naszej symulacji do wyregulowania jest nie 85 klawiszy, tylko 1. Każda jednak najdrobniejsza korekta ustawień wymaga przeprowadzenia nowej symulacji zajmującej kilka minut, dlatego też prawidłowe wyregulowanie wirtualnego mechanizmu zabiera niewiele mniej czasu, co w przypadku prawdziwego instrumentu! Zasadnicza jego zaleta polega jednak na tym, że podobnej regulacji nie trzeba już nigdy powtarzać tak, jak ma to miejsce w przypadku fortepianu.

Normalnie pierwszym krokiem jest wyrównanie klawiatury, tzn. ustalenie klawiszy w takiej pozycji, aby wystawały nad listwą przednią na 20-22 mm. Dokonuje się tego podkładając papierowe podkładki pod klawisz w miejscu jego oparcia, tej czynności możemy jednak zaniechać. Znacznie ważniejszą jest regulacja repetycji. Przede wszystkim należy ustalić pozycję wyjściową bijnika tak, aby tylna powierzchnia bijnika zrównała się dokładnie z pionową osią bródki młotka (baryłki). Dokonuje się tego regulując lewy występ oporowy bijnika (kontakt 3 na rys.3.). W moim modelu korzystam z poprawki „ogr. lewe bij.” oraz „obrót bij.” starając się tak skonfigurować geometrię kontaktu 3, aby w początkowej chwili symulacji odległość pomiędzy ogranicznikiem i bijnikiem była możliwie bliska零, ale nie dochodziło do kontaktu. Takie ustawienie gwarantuje, że w żadnym z kontaktów w układzie nie została „zamrożona” energia mogąca poważnie zaburzyć początkową fazę symulacji polegającą na ustabilizowaniu się całego mechanizmu - zrównoważeniu wszystkich momentów. Na podobnej zasadzie reguluję geometrię większości pozostałych kontaktów. W pierwszej kolejności będzie to kontakt 1 – styk pilota ze stopką. Przy właściwym ustawieniu pilota wykorzystuję poprawkę „wys. pilota”.

Następnie regulujemy wysokość dźwigni repetycyjnej tak, aby czoło bijnika przechodzącego przez otwór dźwigni znajdowało się 0.3 mm niżej górnej powierzchni tej dźwigni. Wykonuje się to obracając śrubę regulacyjną na górnej części figury (kontakt 10). Odpowiada jej poprawka „ogr. górne dźw.” oraz „obrót dźw.”. Kolejnym zabiegiem jest ustawienie młotka w takiej pozycji, aby „opierał” się baryłką na dźwigni repetycyjnej. Wykorzystuję tu poprawkę „obrót młot.”.

Najważniejszym etapem jest regulacja wymyku. Położenie występu oporowego wymyku (pupki) (kontakt 2) reguluje się tak, aby przy powolnym ruchu klawisza uzyskać

3. Model matematyczny.

zatrzymanie młotka w odległości ok. 3 mm od struny (kontakt 11). Służy do tego poprawka „wys.pupki”. Bardzo ważnym jest niedopuszczenie do przylepiania się młotka do struny i odwrotnie, nie powodowanie cięższej gry przez wcześniejsze wyzwolenie.

Następną w kolejności jest regulacja uchylenia klawisza. Wykonuje się ją podobnie, jak wyrównanie klawiatury, wsuwając pod klawisz papierowe podkładki (kontakt 17). Do wykonania tej korekty służy poprawka „dno klaw.” Bardzo ważne jest uzyskanie normalnego uchylenia klawiszy, nie przekraczającego $10 \pm 0,5$ mm zapewniającego przyjemne uderzenie.

Przysparzającą najwięcej trudności jest regulacja chwytnika. Moment hamowania młotka winien następować w czasie, kiedy odpadnie on od struny 12-15 mm w ruchu powrotnym (12). Aby ustawić chwytnik we właściwej pozycji i nadać jego powierzchni odpowiednie pochylenie korzystam z poprawek „odl. chwyt” i „położ. chwyt”.

Często w czasie napraw zdarzają się przypadki ześlizgiwania się młoteczków z chwytników przy słabych uderzeniach. Tak bywa, kiedy skóra na chwytnikach jest zużyta, a ogony rdzeni młotków stały się gładkie aż do połysku na skutek ciągłego tarcia. Taka sytuacja może pojawić się i w modelu, jeśli dobierze się za mały współczynnik tarcia. Jego doświadczalne wyznaczenie obarczone jest duży błędem, dlatego też należy jego wartość zweryfikować na drodze symulacji.

Kolejnym etapem jest regulacja siły i aktywności repetycji. Polega ona na podkręcaniu specjalnej śruby znajdującej się na końcu dźwigni repetycyjnej. Śruba ta naciska na sprężynę repetycyjną wpływając bezpośrednio na aktywność repetycji. Wystarczy wykonać śrubokrętem jeden - dwa obroty, aby przy osłabieniu nacisku palców na opuszczony klawisz sprężyna wyswobadzająca podrzuciła do góry dźwignię repetycyjną razem z leżącym na niej młotkiem. Młotek może być podrzucany nie wyżej niż na 3 mm. Do regulacji naciągu wstępnej sprężyny służy poprawka „spręż.dźw.”. Przy regulacji należy unikać nadmiernej aktywności repetycji, ponieważ przy słabych uderzeniach w klawisz młotek nie jest hamowany przez chwytnik i podrzucany rykoszetuje o strunę.

Do dobrych właściwości repetycyjnych należy zaliczyć nie tylko możliwość dużej liczby powtórzeń uderzeń młotków w struny, ale i maksymalnie szybkie odpadanie młotków od strun. Kiedy dźwignia repetycyjna i jej sprężyna zapewniają młotkowi niezbędną ruchliwość i stan zawieszenia takie ich działanie może również zaszkodzić normalnej pracy mechanizmu, ponieważ po naciśnięciu klawisza młotek otrzymuje uderzenie nie tylko bijnika,

3. Model matematyczny.

ale i dźwigni repetycyjnej, która także po odłączeniu się bijnika będzie dążyć do przyciśnięcia młotka do strun. Żeby zabezpieczyć się przed taką możliwością należy ograniczyć wielkość podniesienia dźwigni repetycyjnej śrubą regulacyjną, wmontowaną w widełki młotka.

Regulację występu oporowego dźwigni repetycyjnej (kontakt 8) przeprowadza się tak, żeby dźwignia repetycyjna zatrzymywała się w czasie ruchu figury o 0,5-1 mm wcześniej, zanim bijnik wysunie się spod bródki młotka (poprawka „ogr. dźw. rep.”). W ten sposób odłączenie dźwigni repetycyjnej zapewnia młotkowi pewną wolną przestrzeń na odsunięcie się od strun.

Ostatnim etapem jest regulacja tłumików. Jest bardzo ważne, aby przy naciśnięciu klawiszy tłumiki podnosili się dopiero w połowie drogi klawiszka i młotka. Wcześniejsze podnoszenie się tłumików znacznie zwiększa „ciężkość” gry. Figury kontrklawiatury lub ich łyżeczki, powinny znajdować się ponad końcami dźwigni klawiszowych o 4-5 mm (kontakt 15). Za normalne należy przyjąć podniesienie tłumików nad strunami (przy całkowitym naciśnięciu klawiszy) o 5-6 mm. Do ustawienia tłumika służą aż trzy poprawki : „dł. pręta tłum.”, „położ. łyżki” i „obrót tłum.”.

Jak widać przy tak dużej liczbie parametrów możliwość wprowadzania łatwych i bezpośrednich korekt do geometrii jest sprawą kluczową w sprawnym wyregulowaniu mechanizmu. Wynika stąd jasno, że identyczne wyregulowanie modelu matematycznego i mechaniki, na jakiej wykonano pomiary doświadczalne, jest niemożliwe. Dlatego też, jak już wspomniałem we wstępie, weryfikacja modelu na podstawie danych doświadczalnych będzie miała charakter głównie jakościowy.

4. Pomiary doświadczalne.

Aby poprawnie zasymulować zderzenia pomiędzy elementami mechanizmu niezbędną jest znajomość właściwości sprężystych materiałów pokrywających powierzchnie kontaktów. W p.3.7. stwierdziłem, że zależności siła reakcji od penetracji opisują właściwości obu ciał będących w kontakcie. Jednak w rzeczywistości odkształcenia drewnianego bijnika, czy metalowego bijnika w porównaniu z odkształceniami filcowej stopki, czy skórzanej baryłki są pomijalne i dlatego wystarczy poddać badaniom jedynie te ostatnie materiały - skórę i filc uwzględniając różną ich grubość i stopień sprasowania w poszczególnych kontaktach¹. W większości kontaktów grubość filcu była na tyle zbliżona, że do ich opisu wystarczał jeden model zbudowany na bazie filcu dna klawiatury. W przypadku stopki, chwytnika i młotka konieczne były szczegółowe badania. W efekcie wykonałem pomiary dla 5-ciu materiałów : filcu dna klawiatury, chwytnika, młotka i stopki oraz dla baryłki

Jak zostało to wcześniej powiedziane, spodziewamy się, że ich właściwości zależą od "historii" zderzenia, tzn. przebiegu prędkości. Dlatego do stworzenia modelu materiału możliwe wiernie opisującego dyssypację energii podczas zderzenia konieczny jest dynamiczny pomiar siły reakcji podczas procesu zderzenia. Przeprowadziłem go wielokrotnie dla różnych materiałów wykorzystując akcelerometr podłączony do komputera i otrzymując w wyniku pętle histerezy siły reakcji od głębokości penetracji [5].

Druga część pomiarów doświadczalnych, dużo bardziej złożona, miała na celu wyznaczenie dynamiki całego procesu uderzenia młoteczka o strunę od chwili naciśnięcia klawisza do jego zwolnienia. Wymagało to zastosowania poza dwoma akcelerometrami czujnika siły. Dodatkowo w celu oszacowania dokładności pomiarów wykorzystałem również kamerę szybkobieżną. Na podstawie analizy otrzymanych z niej obrazów wyznaczyłem trajektorię ruchu elementów mechanizmu, co pozwoliło zweryfikować pomiary przy użyciu akcelerometrów. W rezultacie otrzymałem dane pozwalające ocenić poprawność symulacji.

4.1. Pomiary sprężystości filców i skóry

Aby poznać właściwości sprężyste materiałów, jakimi pokryte są miejsca styku dźwigni mechaniki wykorzystałem do pomiarów masywne wahadło osadzone na łożyskowanej osi. Zaopatrzone je w skalę pozwalającą odczytać początkowe odchylenie kątowe wahadła uderzającego w badany materiał. Notując początkowe i końcowe wychylenie

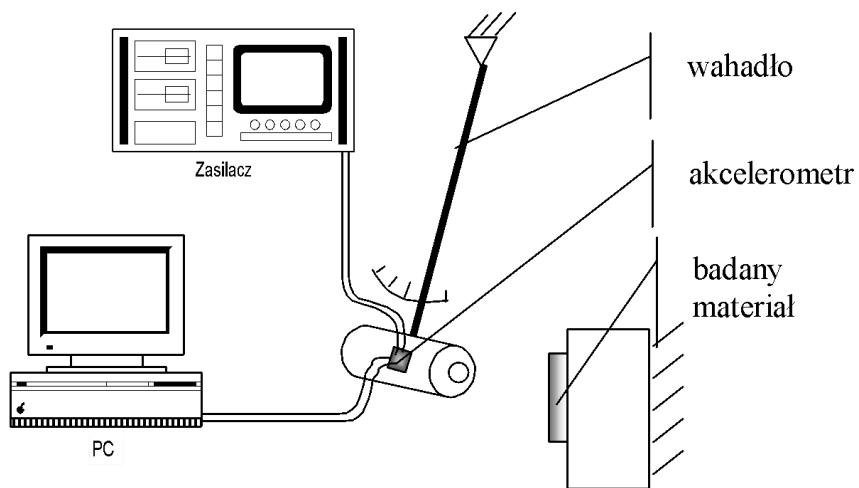
¹ Jedynie dla kontaktu młotka ze struną wykonałem pomiary uwzględniające ugięcia obu ciał.

4. Pomiary doświadczalne.

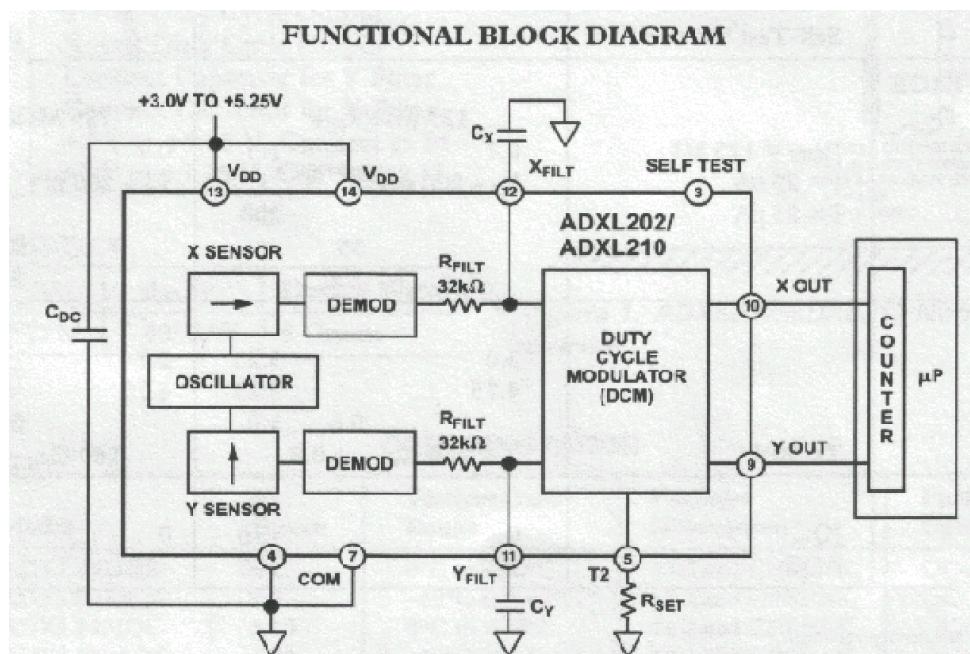
wahadła mogłem obliczyć prędkość na początku zderzenia. Rys 21. przedstawia opisane stanowisko.

Najważniejszym urządzeniem w powyższym stanowisku jest akcelerometr. W moim doświadczeniu użyłem czujniki firmy ANALOG DEVICES - modele ADXL210 i ADXL 150 zapewniające pomiar przyspieszeń w zakresie odpowiednio ± 10 i $\pm 50\text{g}$ w dwóch kierunkach.

Wykorzystywałem naturalnie tylko jeden z nich. Rys. 21. przedstawia schemat blokowy czujnika :

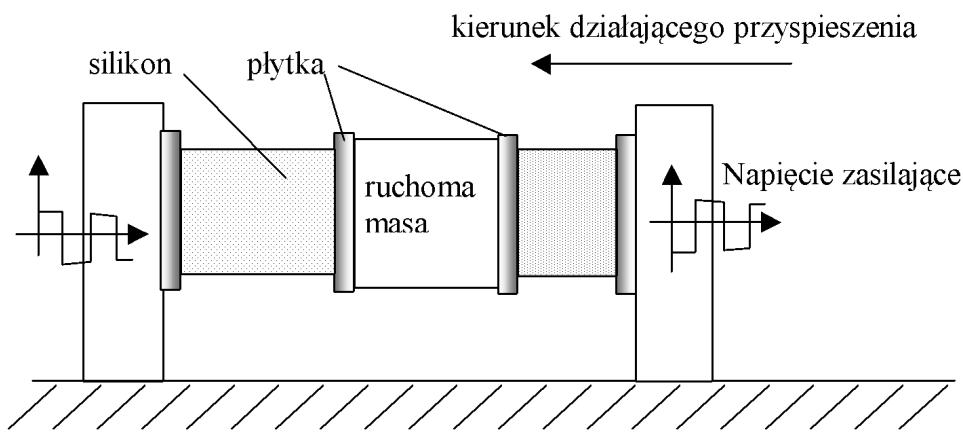


Rys.20. Stanowisko badawcze do pomiarów właściwości sprężystych materiałów.



Rys.21. Schemat blokowy czujnika ADXL210.

Powyższy czujnik działa na zasadzie porównywania pojemności dwóch kondensatorów zawartych pomiędzy dwiema parami płytka oddzielonych od siebie warstwą silikonu stawiającego opór bezwładnościowym siłom dynamicznym związanym z ruchomą masą. Rys. 22. pokazuje ideę działania czujnika:

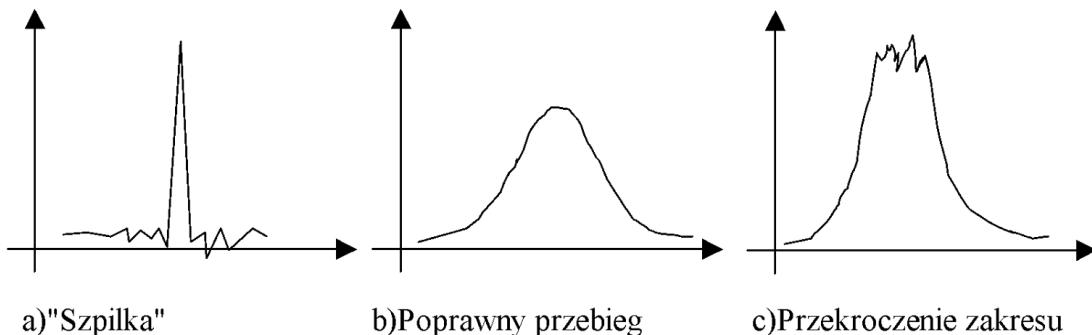


Rys.22. Idea działania akcelerometru

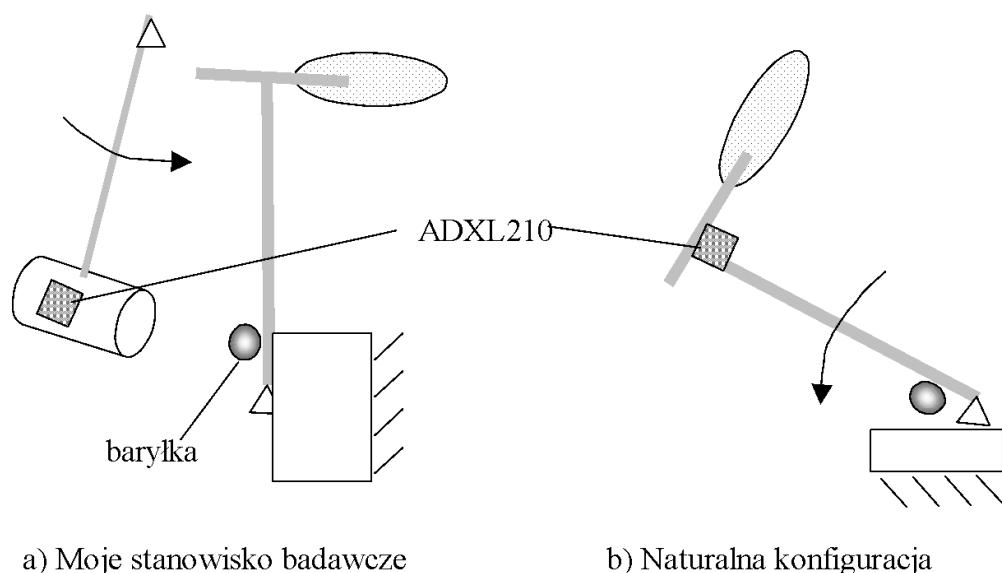
Zewnętrzne płytki przymocowane do poruszającego się ciała zasilane są napięciem zmiennym w formie prostokątnej fali dochodzącej do każdej z nich w przeciwej fazie. Pojawienie się przyspieszenia zaburza równowagę kondensatorów, co objawia się na wyjściu układu prostokątną falą o amplitudzie proporcjonalnej do przyspieszenia. Analiza fazy sygnału wyjściowego pozwala ustalić zwrot przyspieszenia.

Ważną kwestią jest częstotliwość próbkowania sygnału dochodzącego z czujnika. Częste próbkowanie pozwala dokładniej zbadać dynamikę wahadła. Z drugiej jednak strony zwiększa błąd samego akcelerometru. Starałem się tak dobrą częstotliwość próbkowania, aby dokładnie zarejestrować moment zderzenia wahadła z materiałem. Gdyby była ona zbyt mała, zamiast paraboli odwzorowującej przyspieszenie hamujące wahadło, otrzymałbym jedynie skokowo pojawiający się impuls, "szpilkę", na bazie której nie można by niczego oszacować. Po za tym owa parabola musi być na tyle dokładna, aby na podstawie płynnie narastającego i opadającego przebiegu przyspieszeń móc się upewnić, czy nie został przekroczony zakres pomiarowy czujnika. Obrazuje to wyk.7. Właśnie z uwagi na ograniczone możliwości pomiarowe akcelerometru zmuszony byłem przeprowadzić pomiary przy użyciu masywnego wahadła, a nie stosunkowo lekkich elementów mechaniki w ich naturalnej konfiguracji. Zamiast bowiem uderzać np. w baryłkę (skórzany wałeczek na młoteczkę, w który uderza bijnik) wahadłem, mógłbym upuszczać młoteczek i badać właściwości baryłki w "naturalnych" dla niej warunkach pracy - rys.23. Okazuje się jednak,

że dla tak lekkiego ciała, jak młoteczek, przyspieszenia podczas zderzenia baryłki z podłożem dalece wykraczają poza skalę czujników.



Wyk.7. Możliwe przebiegi przyspieszeń podczas zderzeń

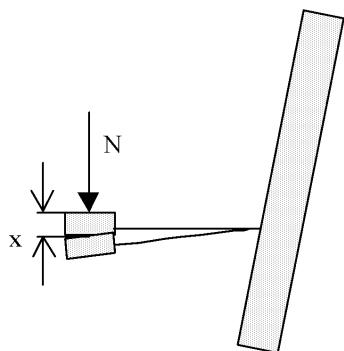


Rys.23. Różne możliwości przeprowadzenia pomiarów.

Analogiczna sytuacja pojawia się przy badaniu filcu stopki (dolnej powierzchni figury, o którą uderza pilot klawisza) jak i filcu, na którym zatrzymuje się klawisz po naciśnięciu - tzw. dno klawiatury. Jedynie zderzenie młotka ze struną z uwagi na jej dużą sprężystość możliwe jest do zarejestrowania w naturalnej konfiguracji.

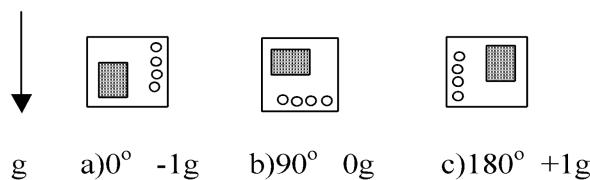
Jak już wcześniej wspomniałem przez sprężystość chwytnika rozumiałem sprężystość całego elementu - klocka pokrytego skórą i osadzonego na sprężystym drucie mającym zasadniczy wpływ na właściwości sprężyste elementu. W efekcie zachodząca tu dyssypacja energii jest trudno mierzalna. Poza tym ponieważ młotek niejako wsuwa się pod chwytnik prędkość zderzenia normalna do powierzchni kontaktu będzie minimalna, dlatego w tym

jednym przypadku wykonanie pomiarów dynamicznych wydaje się zbyteczne. Z powodzeniem można zastąpić je statycznym wyznaczeniem sprężystości obciążając chwytnik różnymi siłami i badając jego ugięcie (rys.24.).

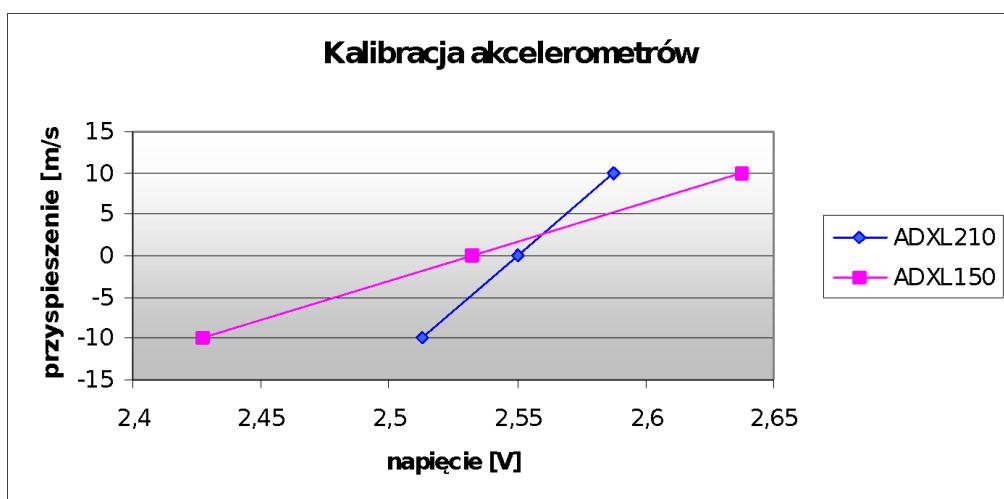


Rys.24. Pomiar sprężystości chwytnika.

Analogowy sygnał wyjściowy z czujnika zawiera się w przedziale 0 - 5V. Dla braku przyspieszenia wynosi ok.2,5V. Zarówno przed, jak i po eksperymencie przeprowadzałem prostą kalibrację czujnika polegającą na sczytaniu z niego sygnału przy różnych jego orientacjach (Rys.25) otrzymując w wyniku liniową charakterystykę napięcie-przyspieszenie czujników (wyk.8).



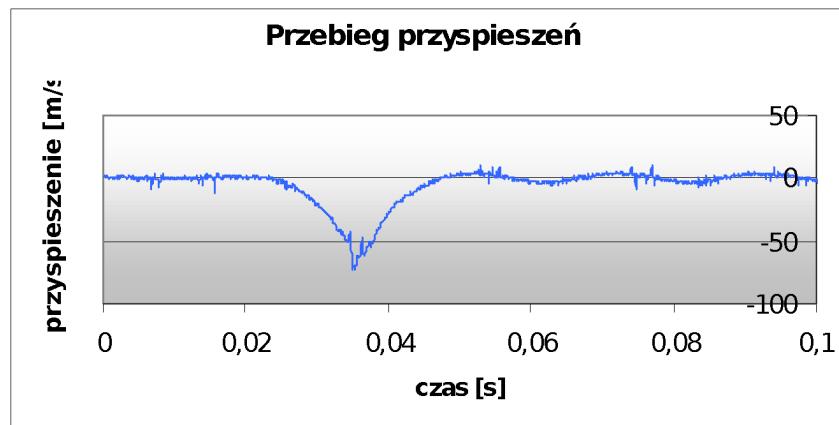
Rys.25. Orientacje czujnika przy kalibracji.



Wyk.8. Liniowe charakterystyki akcelerometrów.

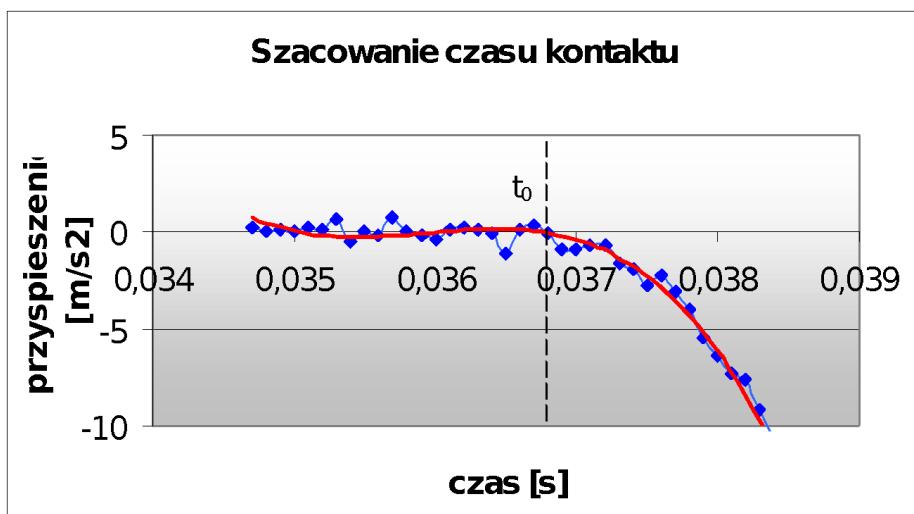
4. Pomary doświadczalne.

Przeprowadziłem po cztery serie pomiarowe dla różnych wychyleń początkowych wahadła dla każdego z czterech materiałów. Otrzymane dane poddałem następnie złożonej "obróbce" przy wykorzystaniu programu MICROSOFT EXCEL oraz MATLAB. Po odjęciu wielkości stałej U_0 od serii i pomnożeniu jej przez współczynnik skali czujnika otrzymany z kalibracji otrzymałem przebieg przyspieszeń. Wyk.9. pokazuje przykładową serię dla baryłki.

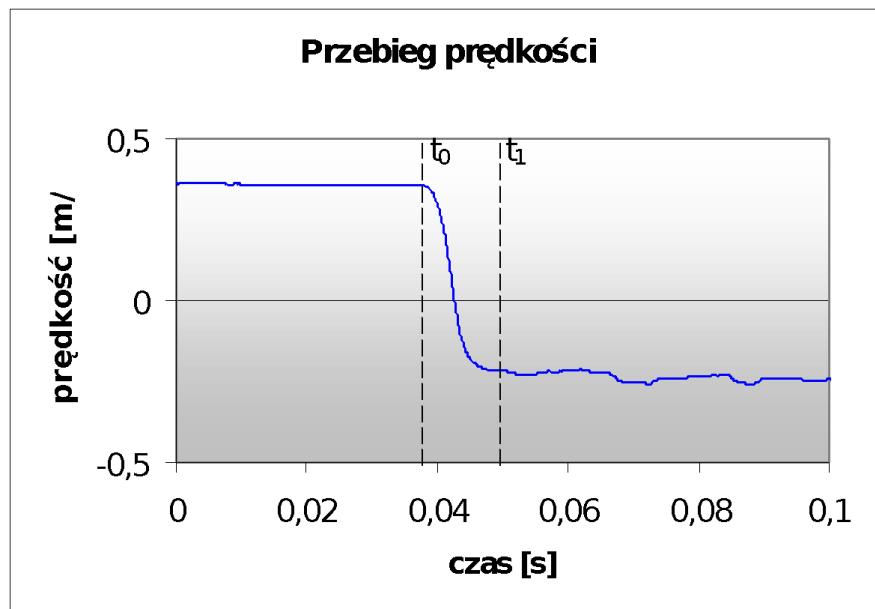


Wyk.9. Przykładowy przebieg przyspieszeń dla baryłki.

Kolejnym krokiem było ustalenie czasu t_0 , w którym następował kontakt pomiędzy wahadłem, a materiałem i czasu t_1 , kiedy ten kontakt zanikał. Dokładnie bowiem w tym przedziale należało scałkować przebieg przyspieszeń przyjmując za prędkość początkową wcześniejszej obliczoną wartość. Aby możliwie dokładnie ustalić czas t_0 i t_1 w okolicy "oderwania" się krzywej przyspieszenia od osi OX aproksymowałem jej przebieg wielomianem, aby ograniczyć błędy wynikające z szumów (Wyk.10.):

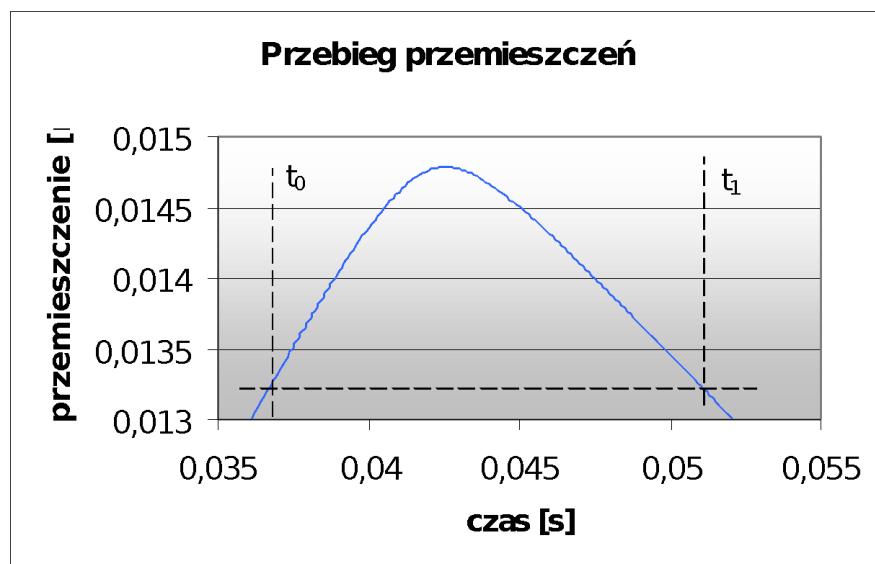


Wyk.10. Szacowanie czasu kontaktu.



Wyk.11. Przykładowy przebieg prędkości dla baryłki.

Ponowne całkowanie przebiegu prędkości (wyk.11.) dało w efekcie przebieg przemieszczeń (Wyk.12.):



Wyk.12. Przykładowy przebieg przemieszczeń baryłki w przedziale czasu $t_0 – t_1$.

Oba całkowania wykonałem metodą prostokątów. Jak widać obecne na wyk.9. zakłócenia sygnału po dwukrotnym całkowaniu niemal całkowicie zanikają.

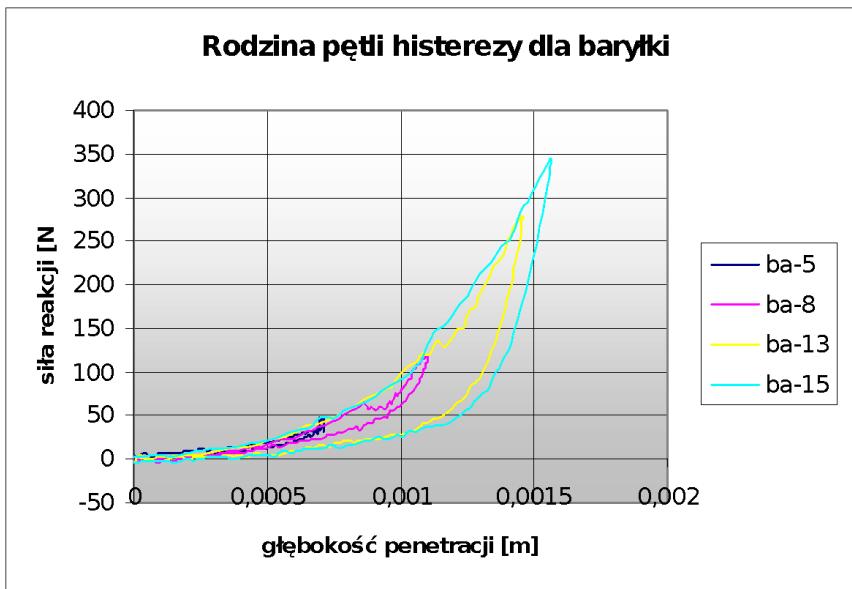
Pozioma przerywana linia na wyk.12. oznacza położenie, od którego mierzona jest penetracja wahadła w głąb materiału. Przeliczając w prosty sposób wartość przyspieszenia (a) na wielkość siły reakcji materiału ($F = a \cdot m$) przy znanej masie wahadła(m), które w chwili zderzenia opisać można jako punkt materialny poruszający się ruchem prostoliniowym,

otrzymujemy poszukiwaną pętlę histerezy siły reakcji od głębokości penetracji (Wyk.13.). Górną krzywą odpowiada fazie hamowania wahadła, dolna - ponownego rozpędzania. Pole pomiędzy krzywymi odpowiada pochłoniętej w trakcie zderzenia energii.

Aby właściwie opisać zachowanie się materiału należy przeanalizować całą rodzinę pętli histerezy dla różnych prędkości zderzenia. Wyk.14. pokazuje taką rodzinę dla wychyleń wahadła kolejno o 5° , 8° , 13° i 15° od położenia równowagi. Wyraźnie na nim widać, że krzywe obciążenia (przy zagłębianiu się wahadła w głąb materiału) dają się opisać jedną krzywą (F_{obc}). Dużo bardziej złożona sytuacja przedstawia się dla krzywych odciążenia (F_{odc}), kiedy wahadło wycofuje się z materiału. Kształt tych krzywych, ich nachylenia do osi OX i położenie w ewidentny sposób zależą od prędkości zderzenia.

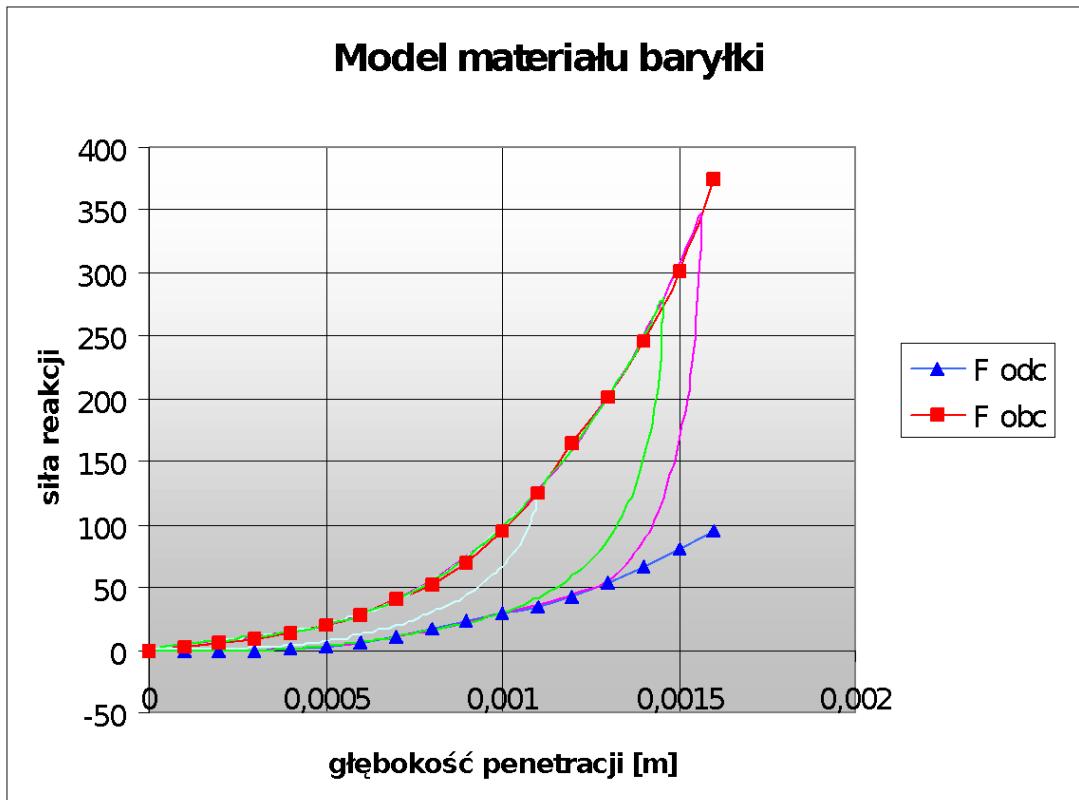


Wyk.13. Przykładowa pętla histerezy siły reakcji od głębokości penetracji dla baryłki



Wyk.14. Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla baryłki

Kształt obu krzywych (F_{obc} i F_{odc}) opisałem dwiema seriami punktów zawartych w tablicy $Mat[1].T$. Przedstawia je Wyk.15., na którym zawarte są również przykładowe pętle histerezy obliczone na bazie powyższych serii i zmierzonych przebiegów penetracji dla odchylenia wahadła odpowiednio o -8° , 13° i 15° .



Wyk.15. Serie danych dla obciążenia i odciążenia oraz zasymulowane pętle histerez dla baryłki.

To, na ile dokładnie model pokrywa się z rzeczywistymi danymi zależy od wielu czynników - przede wszystkim od dokładności wyznaczenia czasu całkowania t_0 i t_1 ; następnie od dokładności wyznaczenia składowej stałej sygnału z czujnika, która to dokładność jest tym mniejsza, im większe jest odchylenie początkowe wahadła. Współczynnik skali czujnika otrzymany z kalibracji wprowadza raczej pomijalne błędy, podobnie jak potraktowanie wahadła jako wahadła matematycznego. Za to istotny wpływ na kształt całej rodziny krzywych ma parametr v_0 w funkcji wagowej. Zawiera się on dla różnych materiałów w przedziale 0.01 –0.02. Dobrałem go metodą prób i błędów, tak, aby zamodelowane powyżej pętle histerezy (wyk.15.) były możliwie najbardziej podobne do zmierzonych w doświadczeniu (wyk.14.). Dodatkowo po przeprowadzeniu właściwej symulacji mechanizmu mechaniki upewniłem się, czy dla oszacowanych w fazie doświadczalnej parametrów v_0 pętle histerezy nie wykazują jakiś nieciągłości. W tym celu w

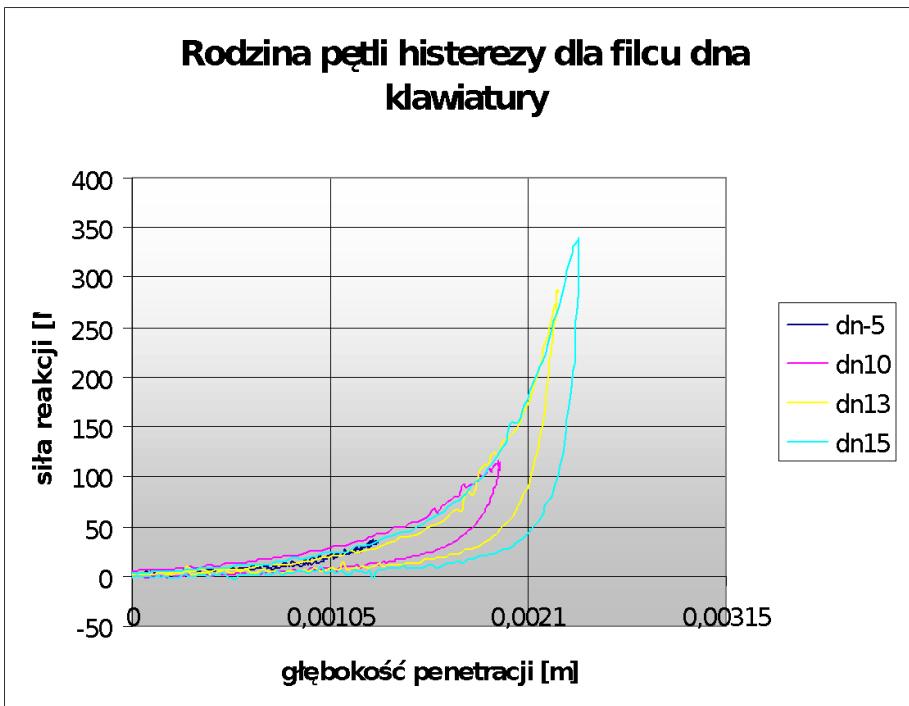
każdym z kontaktów prześledziłem przebieg siły reakcji od penetracji. Ponieważ ruchy elementów mechanizmu są dalece bardziej „zawiłe”, aniżeli ruch wahadła, uzyskane pętle nie będą podobne do tych z wyk.14. Ważne jest jednak, aby nie było w nich gwałtownych przeskoków pomiędzy krzywymi obciążenia i odciążenia. Wyk. 16. przedstawia przykładowy przebieg siły reakcji od penetracji w kontakcie 1 - pilot-stopka. Widać na nim wyraźnie dwie pętle oraz to, że algorytm płynnie przechodzi z krzywej obciążenia do odciążenia, lub też oscyluje gdzieś pomiędzy nimi (dla wagi w : $0 < w < 1$) - dolna część drugiej pętli.



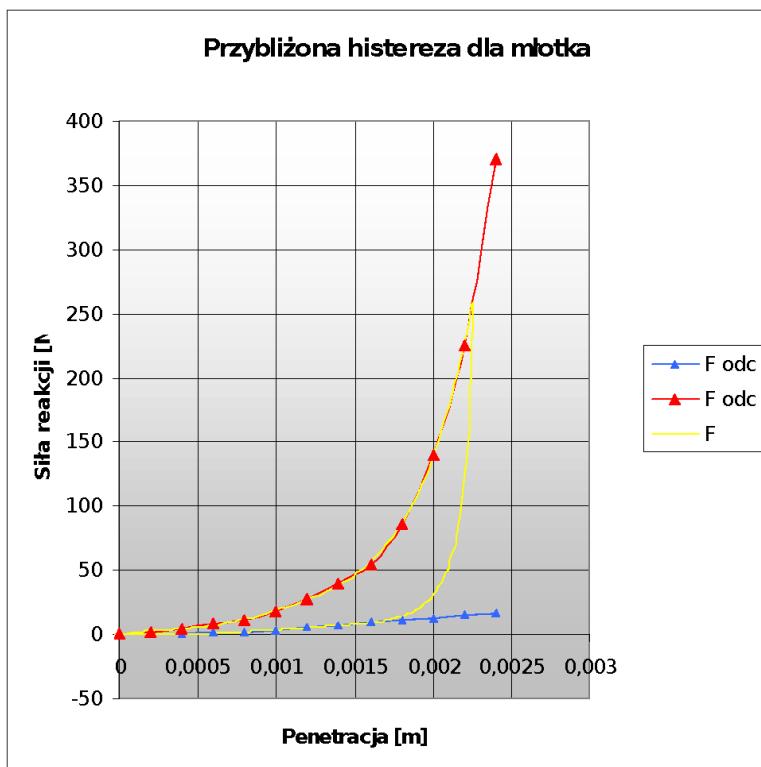
Wyk.16. Przykładowy przebieg siły reakcji od penetracji w kontakcie 1 - pilot-stopka.

Szacunkowa analiza błędów wskazuje, że powyższa metoda pomiarowa na użytku symulacji jest wystarczająco dokładna. Błąd siły reakcji wynikający z dokładności czujnika (wynoszącej ok. $\pm 1\text{m/s}^2$ dla częstotliwości próbkowania 10KHz) i błędu skalowania nie przekracza w sumie 5%. Pełna analiza dokładności pomiaru jest w zasadzie zbyteczna, bowiem błędy stałych materiałowych stanowią tylko jeden z wielu czynników powodujących rozbieżności wyników symulacji i poniżej opisanego doświadczenia. Inne czynniki to błędy wyznaczania momentów bezwładności, położen środków mas ciał, współczynniki tłumienia, tarcia, niedokładności geometrii i co najważniejsze - różnice w regulacji mechanizmów -fizycznego i wirtualnego.

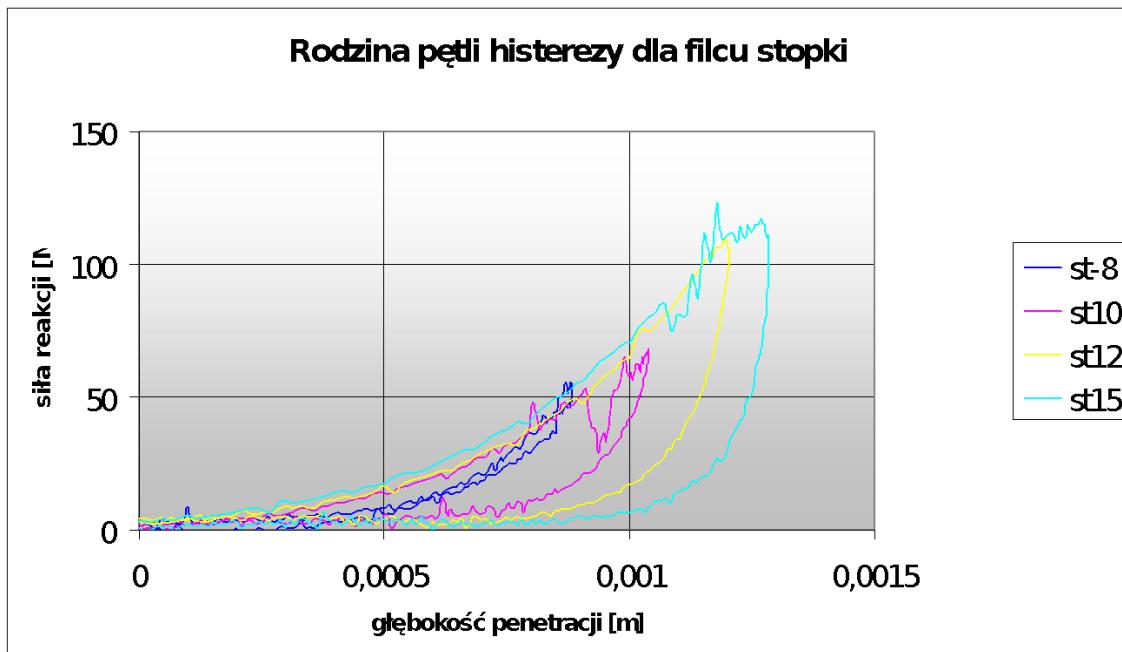
Poniżej zamieszczam wykresy obrazujące właściwości filcu dna klawiatury i filcu stopki (Wyk.17.-.20.) mające podobny charakter, co opisana powyżej baryłka:



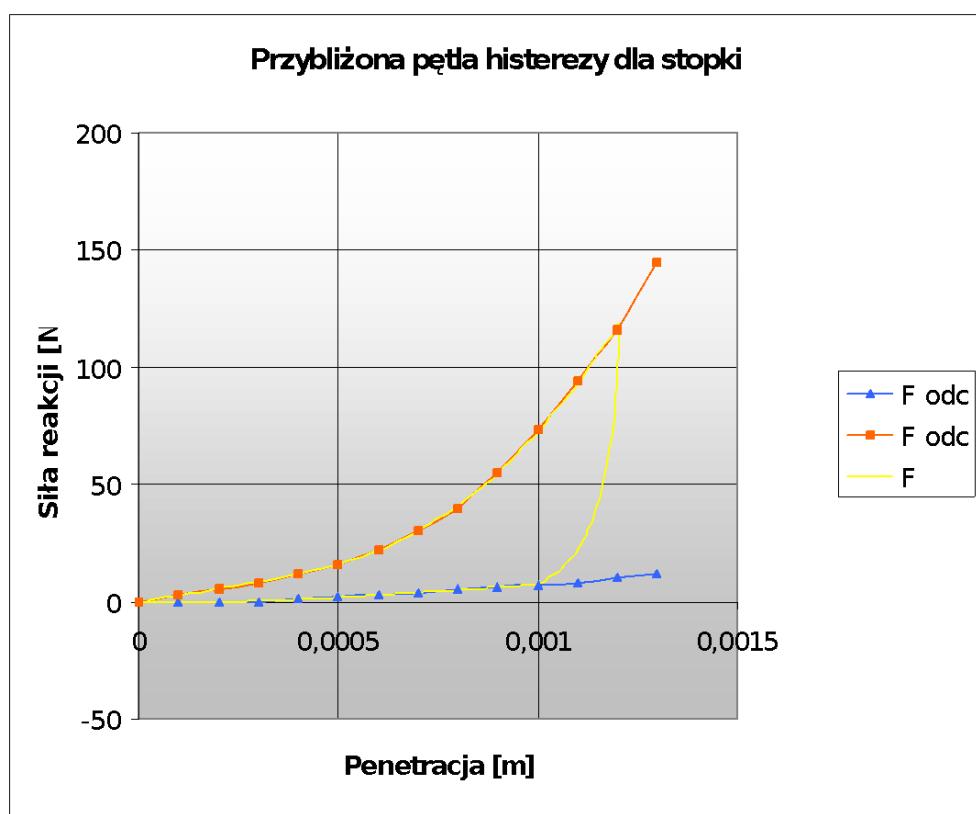
Wyk.17. Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla filcu dna klawiatury. Kąt początkowego wychylenia wahadła $\alpha_0=5^\circ\text{--}15^\circ$.



Wyk.18. Serie danych dla obciążenia i odciążenia oraz zasymulowana pętla histerezy dla filcu dna klawiatury.



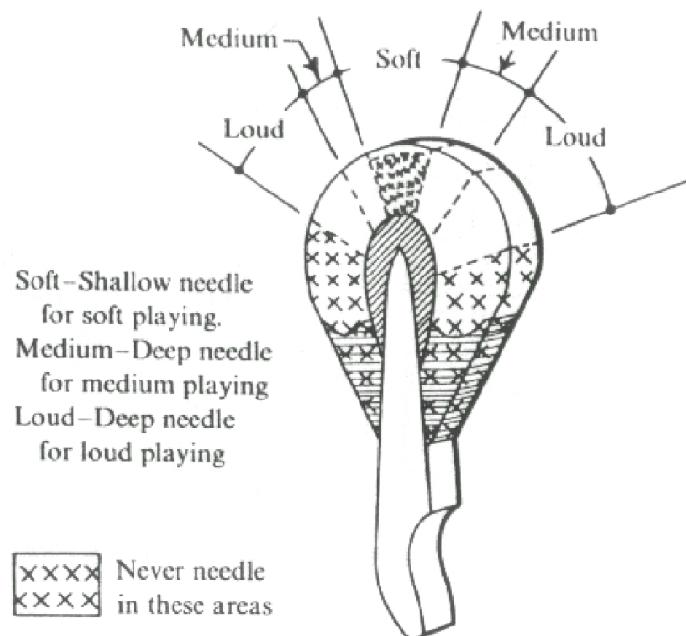
Wyk.19. Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla filcu stopki.
Kąt $\alpha_0=8^\circ\text{--}15^\circ$.



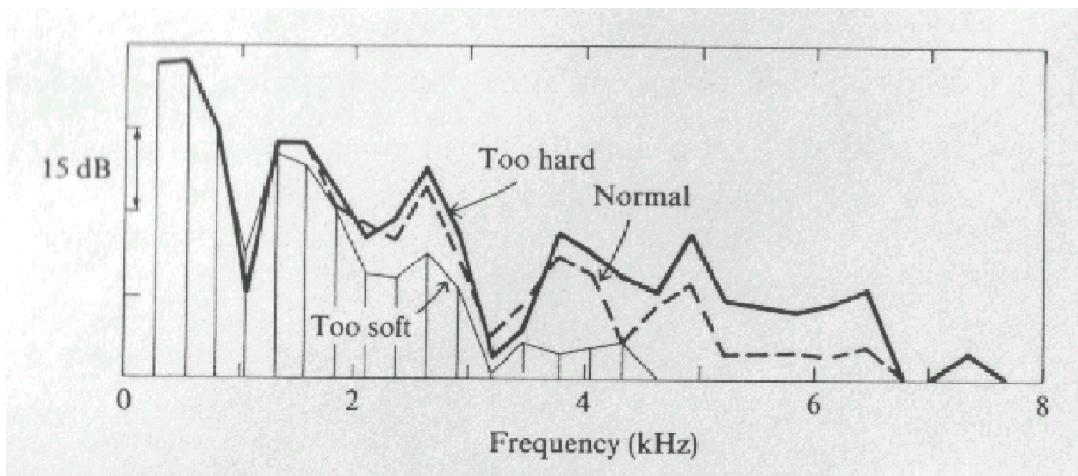
Wyk.20. Serie danych dla obciążenia i odciążenia oraz zasymulowana pętla histerezy dla filcu stopki

Dokładne wyznaczenie właściwości kontaktu młotek –struna w powyższy sposób jest w zasadzie niemożliwy z uwagi na ich niezwykłą złożoność. Filc młoteczka, wyglądający na pierwszy rzut oka nieco niepozornie, jest w rzeczywistości jednym z najważniejszych elementów fortepianu. Od jego właściwości sprężystych zależy w ogromnym stopniu barwa i jakość dźwięku wydobytego z instrumentu. Różnice pomiędzy młoteczkami fortepianów różnych firm (np. YAMAHA, STAINWAY, czy KAWAI) są bardzo trudno mierzalne, mimo to dźwięk wydobywany z tych instrumentów różni się w sposób zasadniczy.

W procesie produkcji młotków wysokiej jakości podczas przyklejania do drewnianego trzonka filc jest naciągany, co wprowadza pewne napięcie wewnętrzne i ma istotny wpływ na jego sprężystość [3]. Twardość młotka zmienia się w zależności od prędkości uderzenia w strunę. Młotek okazuje się twardszy przy grze (fortissimo) aniżeli (pianissimo). Czasem nowy młotek ma zbyt twardą powierzchnię i trzeba go uczynić bardziej miękkim poprzez nakluwanie igłą w odpowiednich miejscach. W zależności, gdzie jest młotek nakluwany, otrzymuje się różną jego twardość (Rys.26.) i w efekcie różną barwę dźwięku (Wyk.21.).

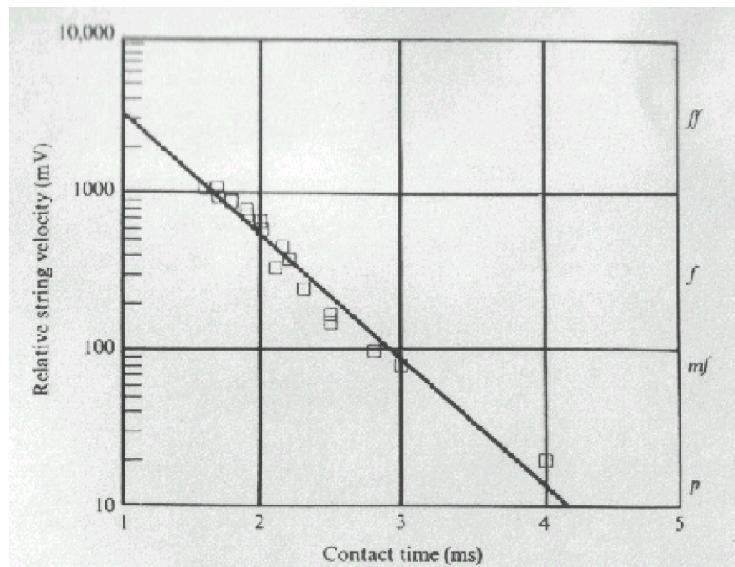


Rys.26. Wpływ miejsca nakluwania młotka na barwę dźwięku.



Wyk.21. Wpływ twardości młotka na widmo akustyczne dźwięku.

Jasne jest zatem, że przy tak złożonych właściwościach sprężystych dokonane przez mnie pomiary sprężystości młoteczka mają charakter jedynie szacunkowy i nie należy się po nich spodziewać zbyt wiele. Ich wiarygodność pomniejsza dodatkowo fakt, że badałem zderzenie młotka nie z prawdziwą struną fortepianową, ale z jej „atrapą” w postaci stalowej linki naciągniętej z dużo mniejszą siłą. Co za tym idzie czas kontaktu młotka ze struną był nieco dłuższy, niż w prawdziwym instrumencie. Wynosił on ok. 8 ms, czyli o ok. 5 ms więcej, niż w prawdziwym fortepianie (wyk. 22.). Dzięki temu przyspieszenia na młoteczkach podczas zderzenia były niższe i mieściły się w zakresie pomiarowym czujnika ADXL150. Tę samą linkę wykorzystałem później przy badaniu dynamiki całej mechaniki, zatem jej naciąg nie powinien wprowadzać rozbieżności pomiędzy pomiarami a wynikami późniejszej symulacji.

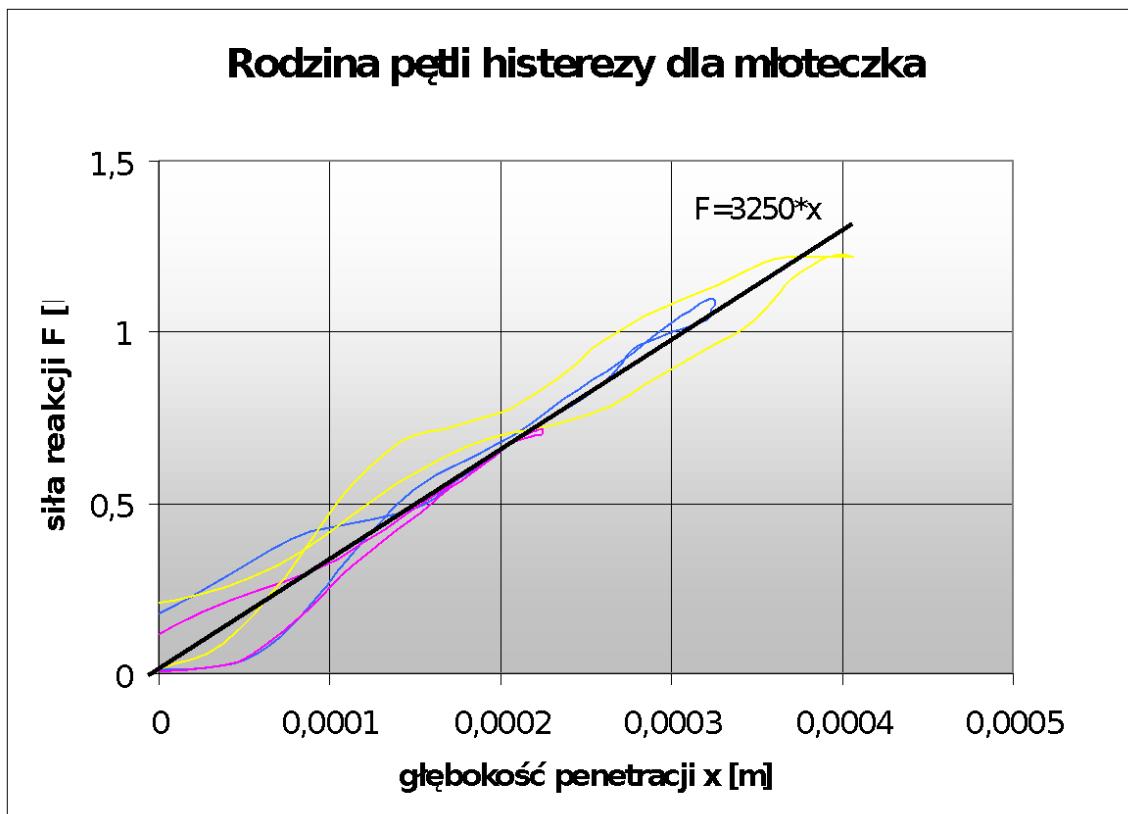


Wyk.22. Przykładowe czasy kontaktu młotka ze struną dla dźwięku C⁴ w zależności od jego głośności [3].

4. Pomiary doświadczalne.

Pomiary przeprowadziłem uderzając w napiętą linkę młoteczkiem z przyklejonym doń czujnikiem. Młotek poruszał się ruchem obrotowym w płaszczyźnie poziomej, co zapewniało stałą wartość zerową napięcia U_0 . W tym przypadku całkowałem przebiegi przyspieszeń nie od chwili zderzenia, ale od momentu, w którym zaczynałem rozpędzać młotek delikatnie trącając go palcem. Cała reszta przebiegała analogicznie do opisanych powyżej działań. Wyk.23. pokazuje rodzinę uzyskanych w ten sposób pętli histerezy siła reakcji od penetracji. Jak widać mają one bardzo trudny do zaproksymowania i zinterpretowania przebieg, jako że w zakresie penetracji 0-0,0001m krzywe odciążenia znajdują się powyżej krzywych obciążenia, w związku z tym sprężystość młotka opisałem jedynie pojedynczą prostą uzależniającą siłę reakcji jedynie od penetracji (x) bez uwzględnienia historii zderzenia:

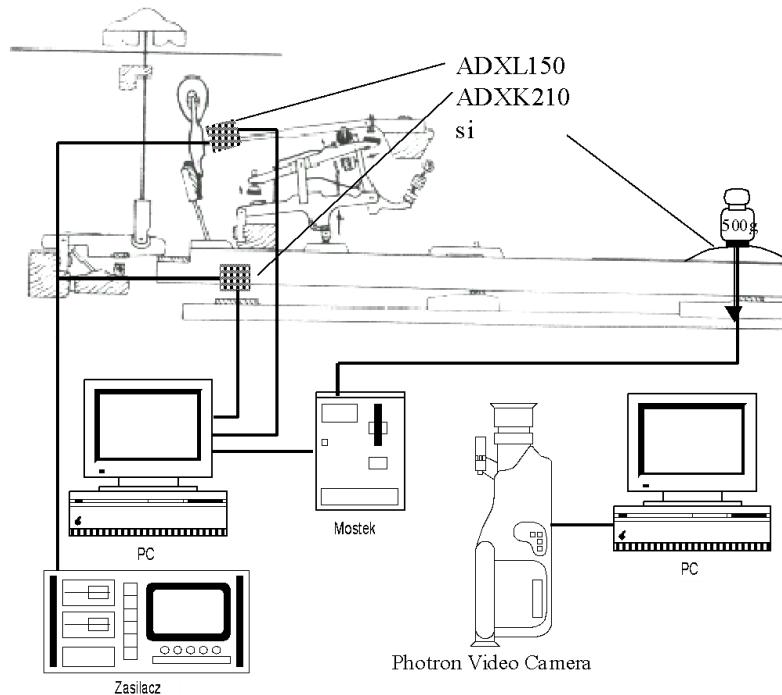
$$F_{\text{reakcji}} = 3250 \cdot x$$



Wyk.23. Rodzina pętli histerezy dla młoteczka i jej aproksymacja liniowa.

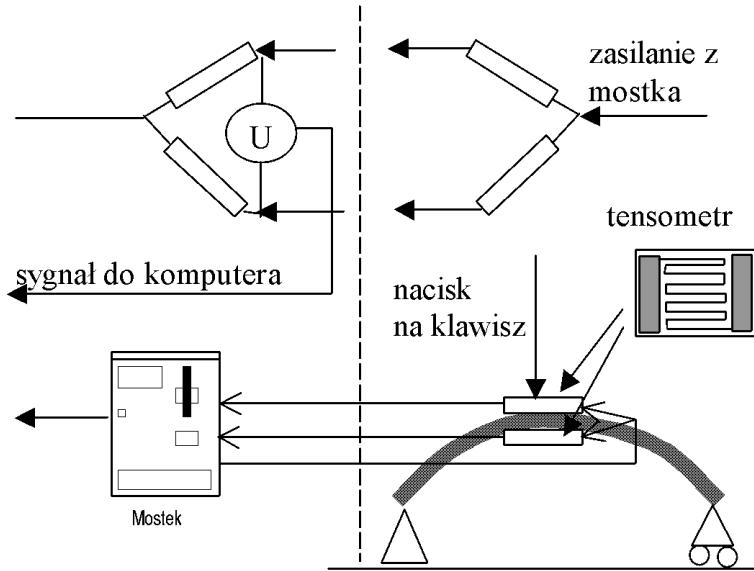
4.2. Pomiary charakterystyk dynamicznych mechaniki fortepianu.

W celu zweryfikowania wyników otrzymanych z symulacji komputerowej niezbędne było zarejestrowanie dynamiki całego procesu uderzenia młoteczka o strunę od chwili naciśnięcia klawisza do jego zwolnienia. Danymi wejściowymi w symulacji są zmieniające się w czasie wartości siły wymuszającej ugięcie klawisza. Danymi wyjściowymi są przebiegi pozycji i prędkości klawisza i młotka. Porównanie tych wielkości z wartościami zmierzonymi eksperymentalnie stanowi główne kryterium oceny poprawności symulacji. Aby zebrać powyższe dane wykorzystałem model mechanizmu młoteczkowego wykonany w skali 1:1. Poza naciągiem struny zastąpionej kawałkiem napiętej metalowej linki w niczym nie odbiega on od standardowego mechanizmu spotykanego w fortepianach. Ponieważ interesowała mnie głównie trajektoria ruchu klawisza i młotka, do nich więc przykleiłem dwa akcelerometry - do klawisza model ADXL 210, a do młotka ADXL 150. Aby zmierzyć siłę wymuszającą ruch klawisza przykleiłem do niego w miejscu kontaktu z palcem prosty czujnik tensometryczny - na tyle lekki, że swoją masą nie zaburzał on dynamiki klawisza. Aby zweryfikować dane otrzymane z akcelerometrów równolegle z pomiarami nagrywałem całą sekwencję na szybkobieżnej kamerze z częstotliwością 250 Hz. Rys.27. przedstawia stanowisko badawcze.:



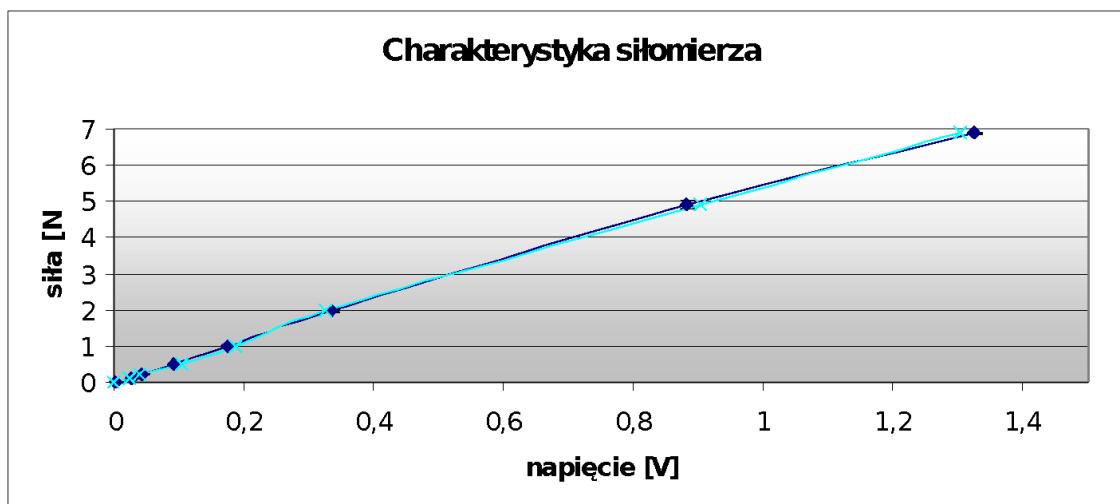
Rys.27. Schemat stanowiska badawczego

Wykorzystany tu czujnik siły zbudowany jest z dwóch tensometrów naklejonych z obu stron na cienką blaszkę pochodząą ze sprężyny zegarkowej. Tensometry podłączone są do mostka firmy ZALMECH w układzie "półmostka" tak, aby wzajemnie się kompensowały (Rys.28.). Sygnał analogowy z mostka trafia następnie do karty analogowo-cyfrowej (model PCL818 firmy ADVANTECH) i razem z sygnałami z akcelerometrów zapisywany jest na dysk.



Rys.28. Schemat działania czujnika tensometrycznego podłączonego do półmostka.

Po dokładnym wyzerowaniu mostka i przeprowadzeniu kalibracji siłomierza przy użyciu odważników o masie od 10g do 700g otrzymałem niemal idealnie liniową charakterystykę siła-napięcie ($F=5,388 U + U_0$) (Wyk.24).

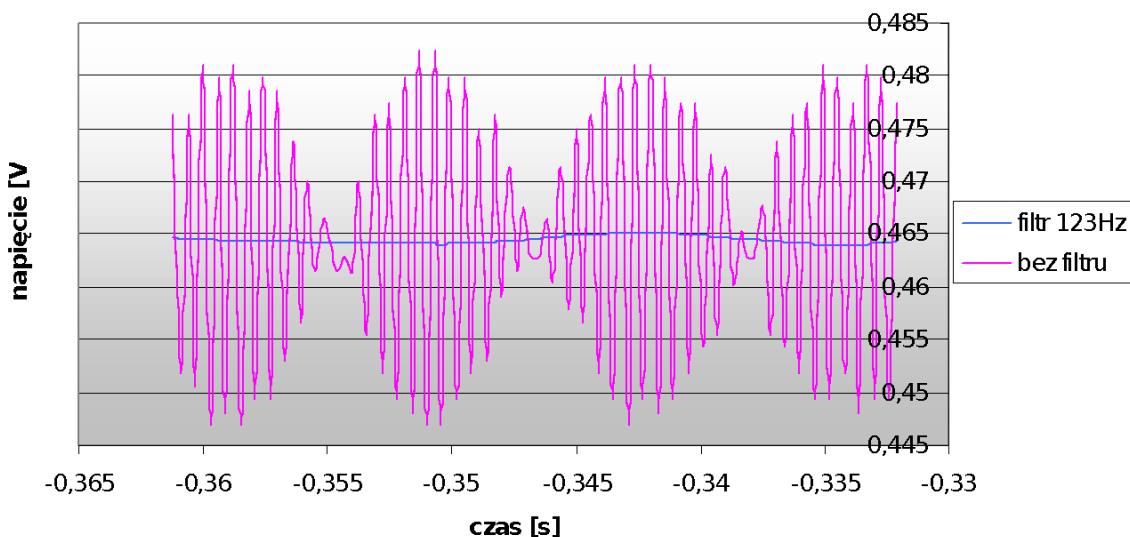


Wyk.24. Charakterystyka siłomierza.

Mimo ekranowania kabli łączących czujnik siły z mostkiem i komputerem sygnał był dość znacznie zanieczyszczony szumami. Pozbywałem się ich filtrując cyfrowo dane z siłomierza. Wykorzystywałem w tym celu procedurę BUTTER (częstotliwość odcięcia 123Hz) w programie MATLAB. Na Wyk.25. pokazany jest sygnał przed i po przefiltrowaniu.

Dokładność pomiaru siły jest trudna do dokładnego ustalenia z uwagi na występujące szумy, zniekształcenia wewnętrz mostka oraz zaburzenia natury termicznej i mechanicznej w tensometrach i blaszce czujnika. Najprościej jest więc oszacować ją na podstawie rozbieżności pomiędzy dwiema seriami kalibracji. Utrzymuję się one na poziomie ok. $\pm 0,1$ N, co stanowi rozsądna wartość błędu czujnika.

Filtrowanie sygnału z siłomierza



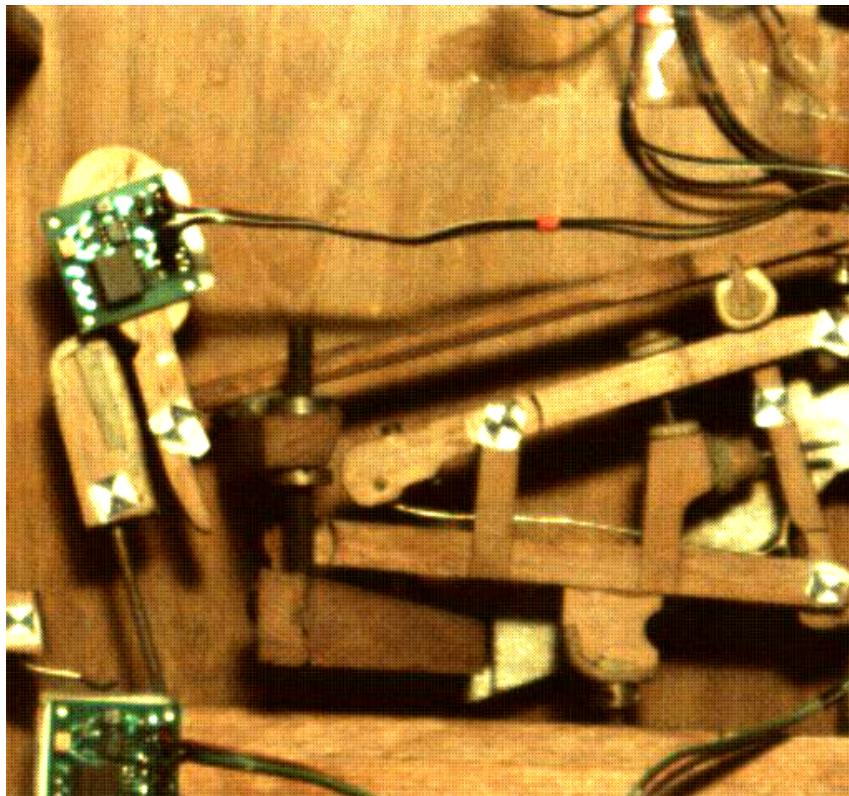
Wyk.25. Filtrowanie sygnału z siłomierza.

Poza czujnikami przyspieszenia i siły w skład stanowiska pomiarowego wchodziła również szybkobieżna kamera video firmy PHATRON - Photron fastcam-pci 1KC - podłączona do drugiego komputera. Zapisywała ona obrazy z częstotliwością 250Hz. Aby dokładniej odczytywać współrzędne elementów po sfilmowaniu w wybranych punktach modelu mechaniki przylepiłem markery widoczne na Rys.29 :

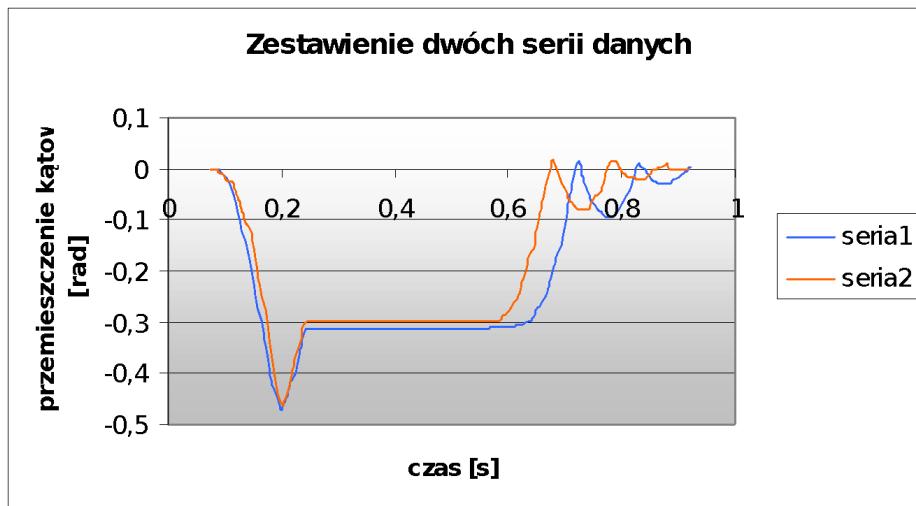
Przeprowadziłem dwie serie pomiarów wymuszając ruch klawisza 500g-owym odważnikiem. Dysponując możliwie powtarzalnym sposobem wymuszenia ruchu chciałem w ten sposób ocenić, na ile cały proces wydobycia dźwięku jest powtarzalny. Okazuje się, że powtarzalność ta na poziomie dokładności moich czujników jest całkowicie zadowalająca. Charakter obu serii przemieszczeń młoteczka na Wyk.26. jest niemal identyczny. Inny jest

4. Pomiarystwo doświadczalne.

jedynie moment powrotu młoteczka na początkowe miejsce, ale nie ma to związku z powtarzalnością zjawiska, a jedynie z czasem, przez jaki utrzymywałem wciśnięty klawisz.



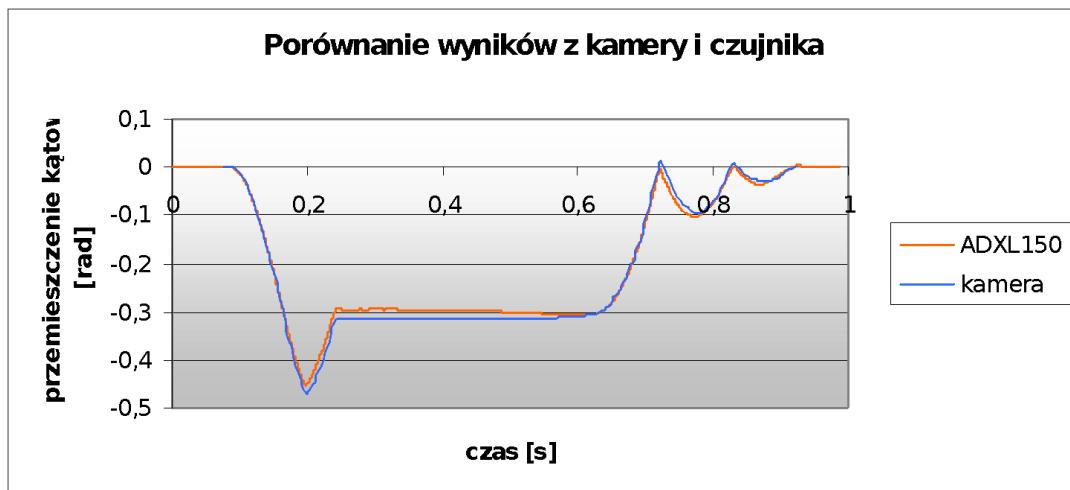
Rys.29. Przykładowa klatka zdjęciowa z widocznymi markerami.



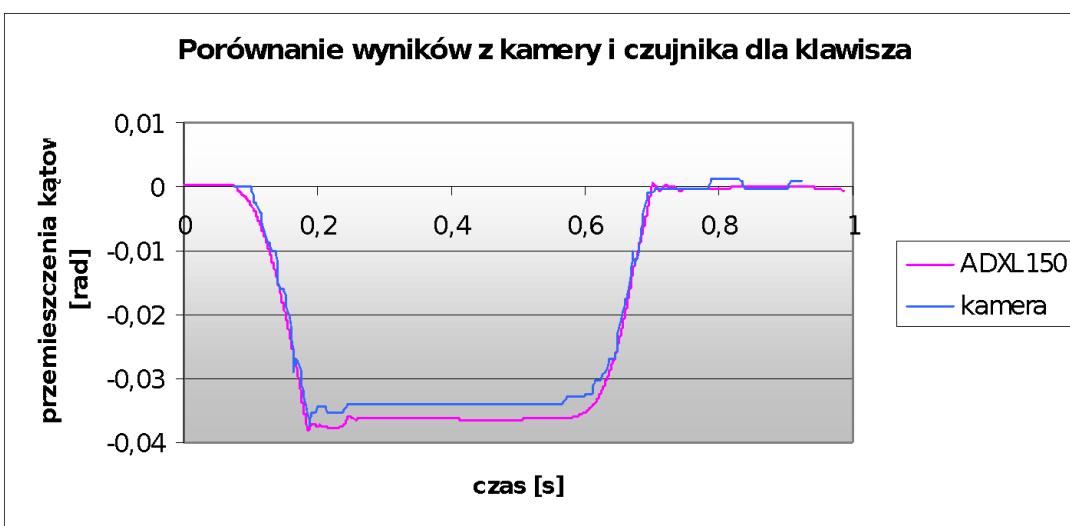
Wyk.26. Zestawienie dwóch serii danych zebranych przez kamerę dla przemieszczeń młoteczka

Dysponując przebiegami przyspieszenia na klawiszu i młotku sczytanymi przez komputer z częstotliwością 3333,3Hz w opisany w p. 4.1. sposób otrzymałem przebiegi przemieszczeń kątowych dla obu tych elementów.

Podobne przebiegi otrzymałem po przeprowadzeniu analizy klatek filmowych. Przy użyciu programu TPS DIG 1.18 sczytałem współrzędne x i y naklejonych na model markerów, a następnie na ich podstawie obliczyłem przemieszczenia kątowe klawisza, młotka, a także pozostałych ruchomych elementów modelu, tzn. figury, tłumika, bijnika i dźwigni repetycyjnej. Na wykresach 27 i 28 zestawiłem wyniki obu metod pomiarowych.



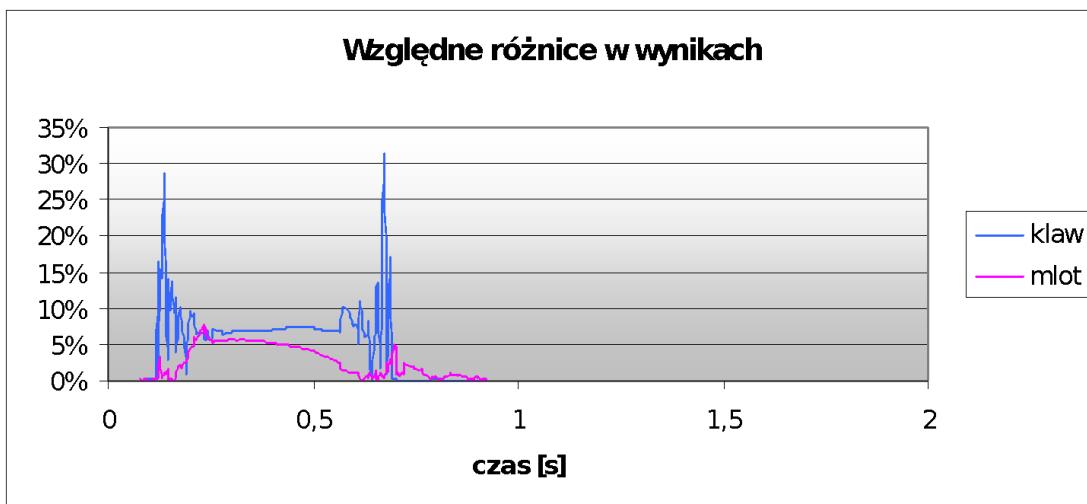
Wyk.27. Porównanie wyników z kamery i czujnika dla przemieszczeń młotka.



Wyk.28. Porównanie wyników z kamery i czujnika dla przemieszczeń klawisza.

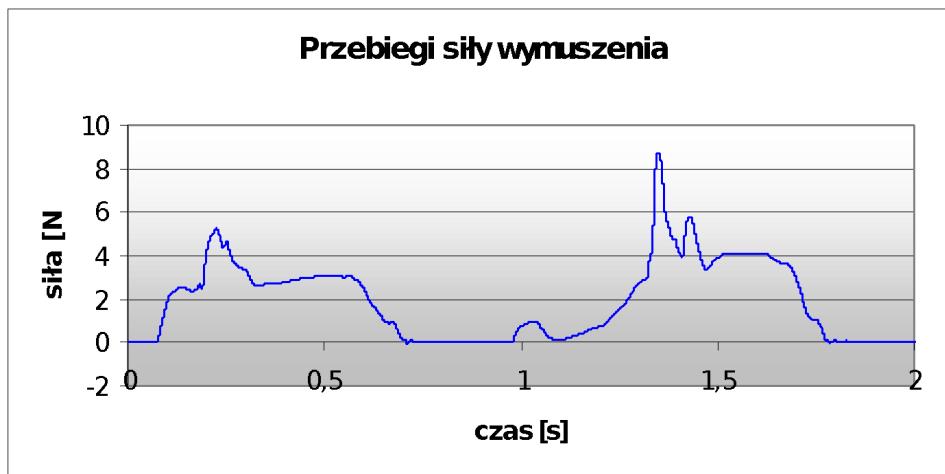
Przyjmując pomiary z kamery za dokładne można na ich podstawie określić dokładność pomiaru za pomocą akcelerometrów. Pojawiające się w nim błędy wskazują czujników i całkowania widoczne na Wyk.29. obrazującym różnice w wynikach otrzymanych z kamery i czujników. W przedziale czasu od ok.0,2-0,6 sek. powyższe błędy utrzymują się dla I serii na poziomie 5-7%. W II serii zawierają się poniżej 10%. Wyższe wartości względnej różnicy poza granicami wspomnianego przedziału wynikają nie z błędów czujnika,

czy całkowania, ale z błędów synchronizacji obu serii danych z kamery i z czujników w czasie. Zarówno zegar w karcie obsługującej kamerę, jak i w karcie analogowo-cyfrowej zbierającej dane z czujników taktują ze skońzoną dokładnością, co jest powodem braku pełnej synchronizacji. I właśnie z tego powodu zmuszony byłem przeprowadzić pomiar na dwa sposoby. Wyniki z kamery, choć dokładniejsze, nie są zsynchronizowane w czasie z przebiegiem siły wymuszającej. Nie możnaby zatem na ich podstawie prawidłowo zweryfikować symulacji, bowiem nie byłyby one zsynchronizowane z zadany w niej wymuszeniem. Wymóg ten spełniają natomiast dane z czujników rejestrowane przez ten sam komputer, co przebieg siły.



Wyk.29. Względne różnice w wynikach z kamery i z czujników dla przemieszczeń klawisza i młotka.

W przypadku pomiaru przebiegu siły wymuszenia działającej na klawisz trudno jest o jakąkolwiek weryfikację. Zakładam, że kalibrację dynamometru wykonaną w warunkach statycznych (wyk.24.) można wykorzystać w warunkach siły szybko zmieniającej się w czasie, jako że bezwładność tensometrów, jak i samego mostka jest tu znikoma. Wyk.30. przedstawia przebiegi sił dla dwóch serii pomiarów. Pojawiające się w drugiej serii znaczne oscylacje siły mogą być skutkiem odbijania się odważnika od blaszki siłomierza na skutek niestartanego jego zwolnienia. Zatem przyjęcie pierwszej serii jako danych wejściowych dla symulacji wydaje się rozsądniejsze.



Wyk.30. Przebiegi sił wymuszenia dla dwóch serii danych.

Ostatnim etapem fazy doświadczalnej było zważenie elementów i wyznaczenie ich momentów bezwładności I. W tym celu zmierzyłem dla każdego ciała okres T, z jakim wykonywało ono ruch wahadłowy (30), a następnie korzystając z równania ruchu :

$$I\alpha''(t) = -Q \sin \alpha(t) r$$

gdzie : Q – ciężar ciała; r – ramię ciężaru

uproszczenia (przyjmowanego dla małych kątów):

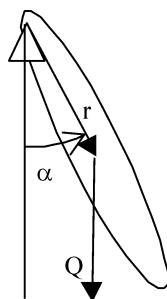
$$\sin \alpha(t) \approx \alpha(t)$$

oraz postaci drgań harmonicznych :

$$\alpha(t) = A \sin(2\pi/T * t)$$

wyznaczyłem wzór na moment bezwładności :

$$I = \frac{QrT^2}{4\pi^2}$$



Rys.30. Ruch wahadłowy ciała.

5. Weryfikacja symulacji.

Jak już zaznaczyłem we wstępie pracy poniższa weryfikacja ma charakter bardziej jakościowy, aniżeli ilościowy. Pierwszym jej etapem jest ustalenie, czy zadana w symulacji dokładność całkowania numerycznego jest wystarczająca. W tym celu przeprowadziłem trzy serie obliczeń dla tolerancji przedstawionych w tab.5. Zawiera ona również maksymalne różnice w obliczonych prędkościach młoteczka pomiędzy kolejnymi seriami (1. i 2. oraz 2. i 3.) (wyk.31. i 32.). Powyższą weryfikację przeprowadzam na bazie prędkości młoteczka, bo wielkość ta określając czułość mechanizmu ma w całej symulacji największe znaczenie. Stanowi niejako syntezę informacji o zjawiskach zachodzących w układzie.

Numer serii	Tolerancja		Max. różnice pomiędzy seriami
	Względna	Bezwzględna	
1.	1e-5	1e-5	0,865
2.	1e-6	1e-6	
3.	1e-7	1e-7	0,074

Tab.5. Porównanie wyników dla różnej dokładności całkowania.



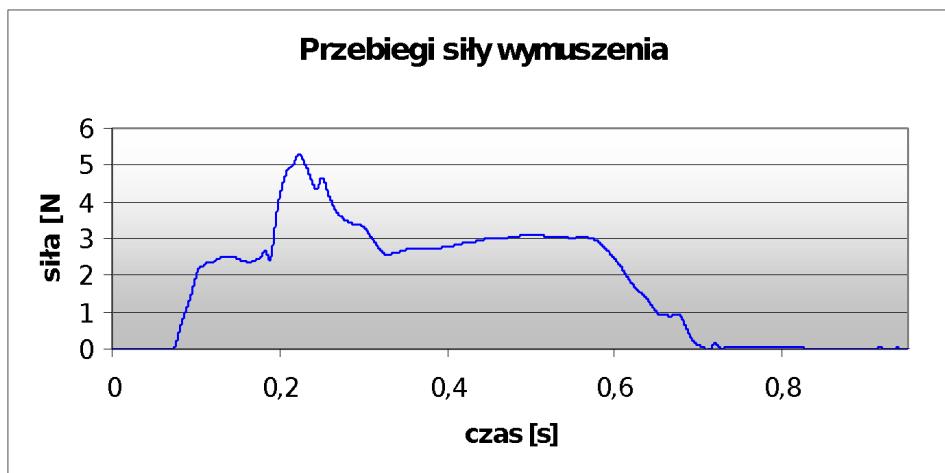
Wyk.31. Porównanie prędkości młoteczka dla różnej dokładności całkowania (seria 1. i 2.).



Wyk.32. Porównanie prędkości mloteczka dla różnej dokładności całkowania (seria 2. i 3.).

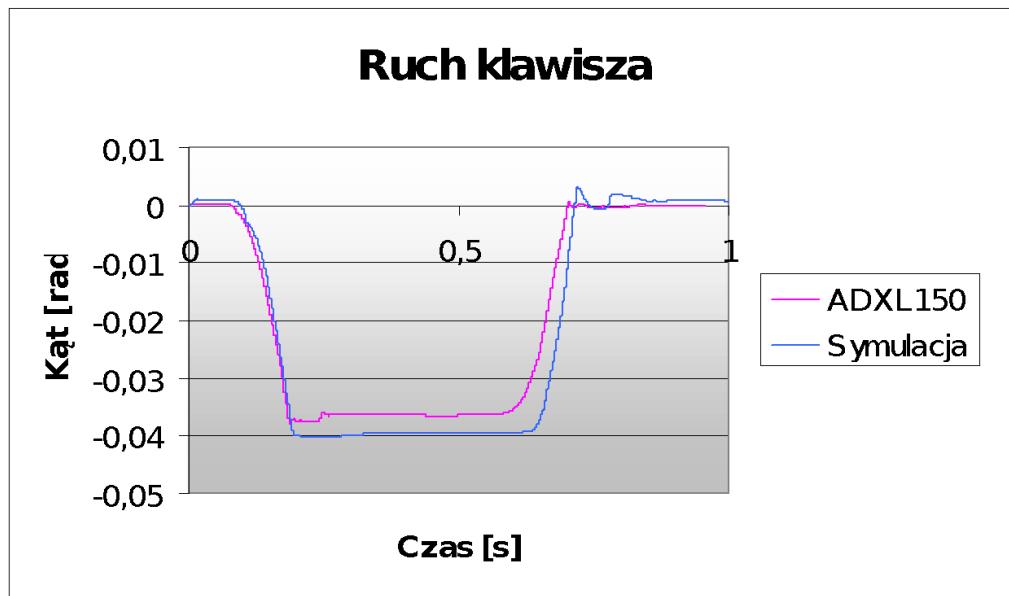
Jak widać dla dokładności $1e-6$ i większej wyniki praktycznie się nie różnią, co widać na wyk.32., gdzie oba przebiegi niemal zupełnie się pokrywają. Wynika stąd wniosek, że tolerancja $1e-6$ jest wystarczająca i serię 2. można uznać za ostateczny wynik symulacji.

Najprostsza forma weryfikacji polega na porównaniu wyników obliczeń i doświadczenia dla jednakoowego wymuszenia (wyk.33). Zestawienie to ujęte jest na wykresach wyk.34.,35. i wyk.37.,38.)



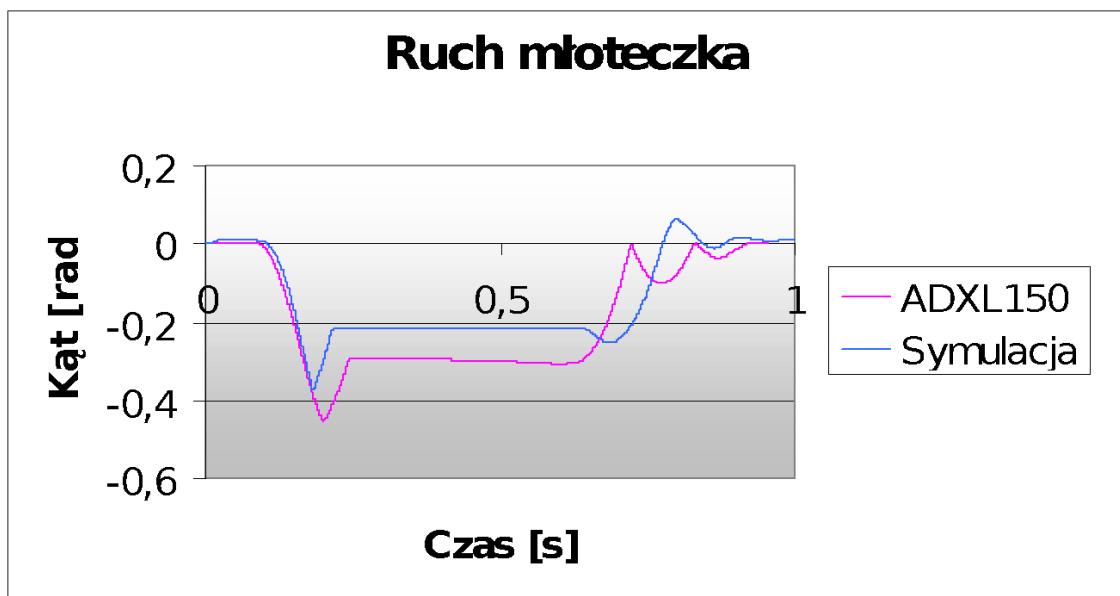
Wyk.33. Wykorzystany w symulacji przebieg siły wymuszenia działającej na klawisz.

Jak widać wymuszenie pojawia się dopiero po upływie ok. 0,075 sek. od startu pomiarów. Opóźnienie to pozwala symulowanej mechanice na osiągnięcie zrównoważenia. Jak bowiem wspomniałem w rozdziale 4. większość kontaktów ustawiiona jest na granicy styku, kiedy nie pojawiają się jeszcze żadne siły reakcji. Gwałtowny skok wymuszenia jest konsekwencją wyhamowania odważnika ustawnionego na siłomierzu (rys.27.) w momencie opadnięcia klawisza na dno klawiatury (po czasie ok.0,2 sek.).



Wyk.34. Porównanie wyników symulacji i doświadczenia dla przemieszczeń klawisza.

Widoczne różnice w uchyleniu wynikają z różnic w regulacji filcu dna klawiatury. Optymalne ugięcie liczone w rad wynosi $-0,038 \pm 0,002$ rad, zatem oba wyniki mieszczą się w przedziale tolerancji. Drobne oscylacje klawisza pojawiające się w symulacji wynikają zapewne ze zbyt małego współczynnika tłumienia ruchu klawisza. Jego wyznaczenie jest bardzo trudne, ponieważ środek ciężkości klawisza niemal pokrywa się z osią obrotu. Dlatego nie można go wprowadzić w swobodny ruch wahadłowy w celu wyznaczenia tłumienia tak, aby obracał się na pionowym kołku (rys.1).

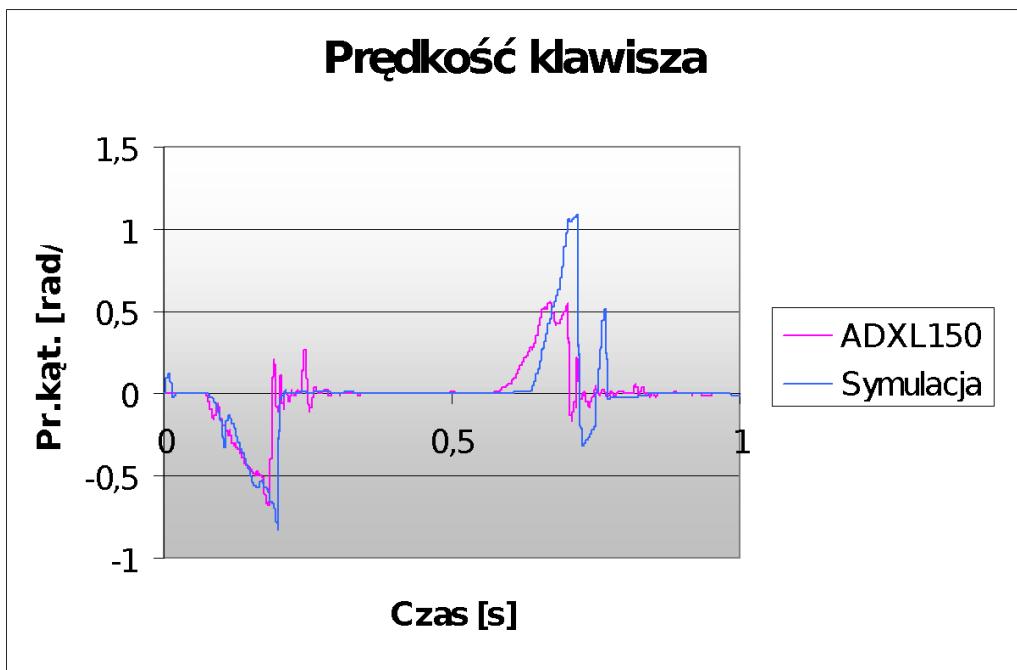


Wyk.35. Porównanie wyników symulacji i doświadczenia dla przemieszczeń młotka.

Dość znaczne różnice na wyk. 35. mają swoje źródło nie tylko w odmiennej regulacji modelu fizycznego i wirtualnego (głównie ustawienie chwytnika), ale również w innej ich geometrii. Wynika stąd, że w fazie doświadczalnej struna umieszczona była cały o centymetr wyżej i co za tym idzie skok młotka był większy o ok.0,03 rad. Jednocześnie chwytnik był bliżej młotka i wcześniej go zatrzymywał, zanim ten zdążył ugiąć dźwignię repetycyjną. Dlatego w doświadczeniu młotek nie jest podbijany przez dźwignię. W [2] zaleca się jedynie, aby to podbicie nie przekraczało 3mm, co zachowane jest w symulacji (podbicie pojawia się na wyk.35. po upływie ok. 0,7 sek. jako niewielki „pagórek”). Nie jest ono jednak konieczne do prawidłowej pracy mechanizmu i może w ogóle nie wystąpić. Zatem pomimo tak znaczących różnic oba modele spełniają normy regulacji. W symulowanym modelu umyślnie tak ustawiłem chwytnik, aby nastąpiło ugięcie dźwigni rep. Dowodzi to, że algorytm poprawnie modeluje zjawisko tarcia i możliwe jest w nim „zamrożenie” energii potencjalnej sprężyny. Choć nie wydaje się to specjalnie skomplikowane, zamodelowanie owego „zamrożenia” przysporzyło mi w trakcie pisania programu najwięcej trudności. Objawia się tu w pełni problem przejścia tarcia dynamicznego (T_d) w statyczne (T_s) i związana z tym zmiana zwrotu siły tarcia (T). Aby młotek nie wyślizgiwał się spod unieruchamiającego go chwytnika przejście to musi odbywać się płynnie i w sposób zdecydowany (waga w_d powinna przybierać wartości możliwie skrajne). Poniższe wykresy (wyk.36.) obrazują cały ten proces. Wyraźnie widać, że funkcja wagowa zmienia się w sposób płynny nie powodując nieciągłości. Jednocześnie zahamowanie – to, co najtrudniej było zamodelować - jest skuteczne i trwałe. Siła tarcia po zatrzymaniu się młotka przybiera stałą wartość, co świadczy o stabilności modelu. Dzięki temu prędkość młotka po zetknięciu z chwytnikiem całkowicie zanika – nie występują poślizgi, co widać na wyk. 38.



Wyk.36. Przebieg procesu wyhamowania młotka przez chwytnik.



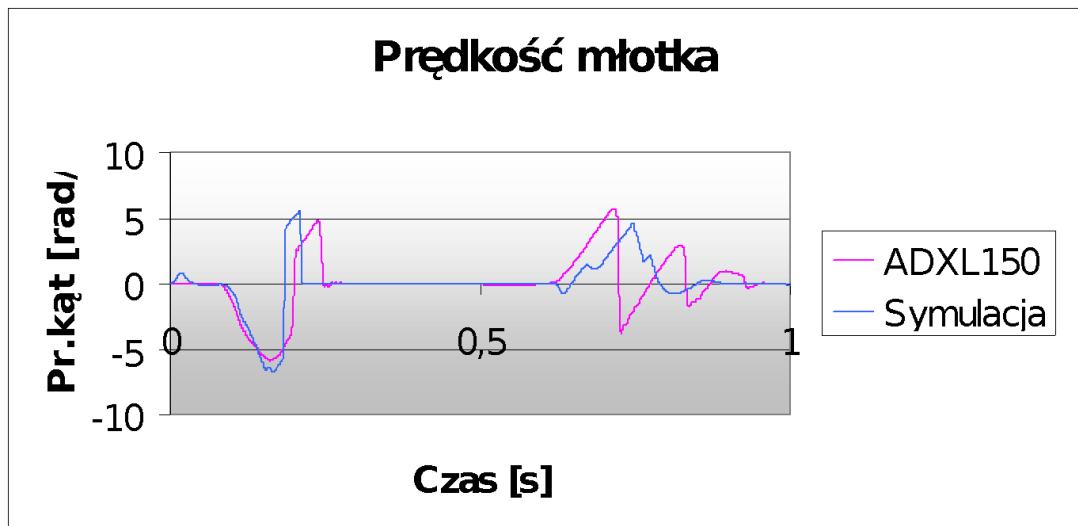
Wyk.37. Porównanie wyników symulacji i doświadczenia dla prędkości klawisza.

Rozbieżności na wyk.37. są w zasadzie znikome. Przyrost prędkości w początkowej fazie wymuszenia (do ok. 0,2 sek.) jest zdumiewająco podobny. Świadczy to o tym, że oba modele stawiają niemal identyczny opór! Ponieważ wielkość ta (opór klawisza) w sposób minimalny zależy od regulacji – czynnika najbardziej utrudniającego weryfikację – pokrycie się jej wartości w obu modelach stanowi wiarygodne potwierdzenie stosunkowo dużej dokładności symulacji. Opór klawisza zależy w głównej mierze od geometrii, mas i momentów bezwładności elementów mechaniki. Regulacja może co najwyżej przyspieszyć lub opóźnić pewne zjawiska, jak np. wyzwolenie, ale nie wpływa na powyższe wielkości, a co za tym idzie i na sam opór.

Jak wcześniej wspomniałem, wielkością w sposób zasadniczy weryfikującą symulację jest prędkość młoteczka, w szczególności prędkość tuż przed uderzeniem w strunę. Jak wynika z wyk.38. względna różnica dla tego momentu wynosi zaledwie 12%. Jest to niewielka wartość zważywszy na to, jak wiele czynników wprowadza rozbieżności pomiędzy doświadczeniem, a symulacją.

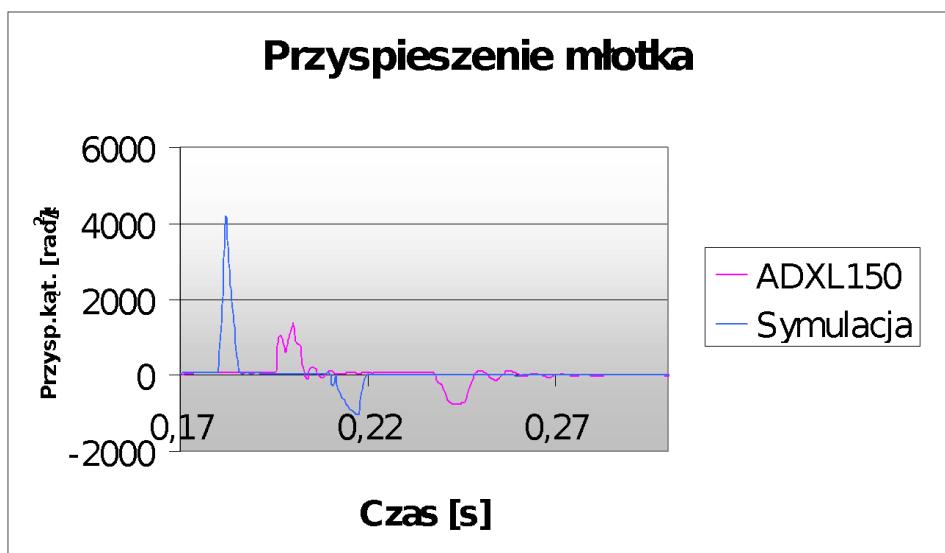
Wyk.39. pokazuje przyspieszenia, jakim poddany jest młotek podczas kontaktu ze struną. W symulacji odbicie młotka przebiega znacznie gwałtowniej – przyspieszenie jest większe. Świadczy to o nieprawidłowo wyznaczonej pętli histerezy dla kontaktu młotek-struna. Jak zaznaczyłem w roz.4. wyniki doświadczalne nie były w tym przypadku specjalnie przekonujące (krzywa odciążenia przekraczała w pewnym miejscu krzywą

obciążenia). Dlatego też wyznaczone krzywe (w szczególności odciążenia) obarczone są dużym błędem. Ponieważ jednak użyta struna była tylko atrapą – metalową linką napiętą na tyle słabo, aby w ogóle dało się przeprowadzić pomiary, nie należy przywiązywać specjalnej uwagi do kwestii właściwości sprężystych w kontakcie młotek- struna.



Wyk.38. Porównanie wyników symulacji i doświadczenia dla prędkości młotka.

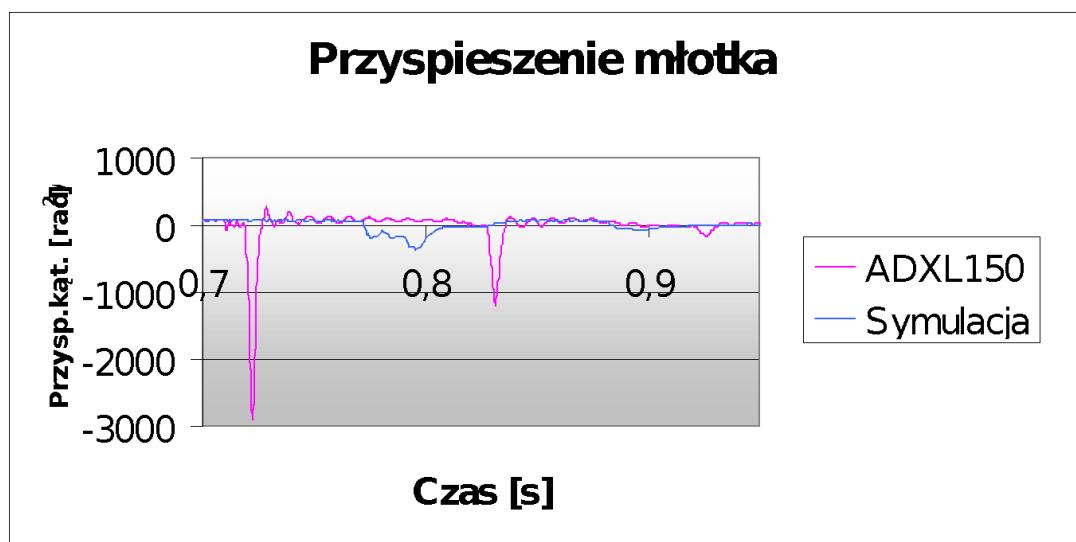
W zasadzie najprościej byłoby ustalić te właściwości tak dobierając krzywe obciążenia i odciążenia, aby w doświadczeniu i symulacji zgadzał się czas kontaktu młotka ze struną – druga po prędkości młotka najważniejsza wielkość w całej symulacji. Ma ona bowiem zasadniczy wpływ na zagadnienia natury akustycznej istotne dla pianisty w stopniu równym, jeśli nie większym, niż np. opór klawiatury – wielkość typowo mechaniczna.



Wyk.39. Porównanie wyników symulacji i doświadczenia dla przyspieszeń młotka przy uderzeniu w strunę.

Różnice w przyspieszeniach młotka pomiędzy doświadczeniem, a symulacją wpływają bezpośrednio na rozbieżności w czasie powyższego kontaktu. I tak w doświadczeniu wynosi on 8ms, a w symulacji - 5ms. Jak podaje [3] normalnie waha się on w granicach 2–4ms. To, że w symulacji odbicie od struny następuje wcześniej, jest oczywiście konsekwencją niższego położenia struny.

Kolejną kwestią wartą komentarza jest sposób opadnięcia młotka na figurę już po zwolnieniu klawisza. W doświadczeniu młotek wyraźnie odbija się kilka razy od dźwigni, bądź bijnika, zanim ostatecznie się zatrzyma. W symulacji młotek przed zatrzymaniem znacznie „zagłębia się” w figurze (wyk.34), podobnie, jak stopka figury „zagłębia się” w pilocie klawisza (wyk.35). Przyczyna tak znacznej penetracji leży zapewne w zbyt miękkich zamodelowanych materiałach. Widoczne jest to również w postaci mniejszych i łagodniejszych przyspieszeń (wyk.40). Ponieważ jednak opisana tu faza ruchu młoteczka nie posiada większego wpływu na poprawność działania mechanizmu, opisane zjawisko ma dla nas znikome znaczenie.



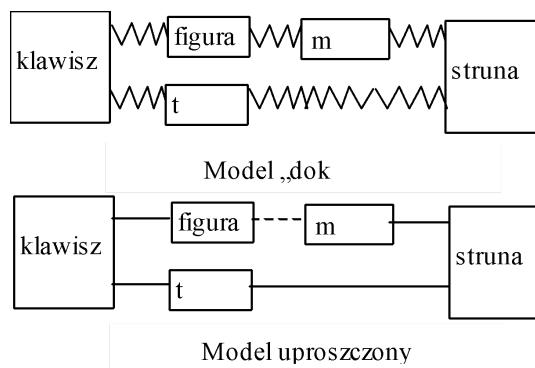
Wyk.40. Porównanie wyników symulacji i doświadczenia dla prędkości młotka przy podniesieniu się klawisza.

Podsumowując powyższą weryfikację należy stwierdzić, że pomimo wielu rozbieżności wynikających z różnej regulacji, czy też trudności w ustaleniu właściwości sprężystych ciał, najważniejsze wielkości, tzn. opór klawisza i prędkość młotka w chwili uderzenia w strunę są bardzo zbliżone, co potwierdza wiarygodność symulacji. Jej wyniki mogłyby być naturalnie dokładniejsze, ale na użytek ogólnej analizy fortepianu, a nie konkretnego modelu, czy marki wydają się one w pełni satysfakcjonujące.

6. Model uproszczony

W niniejszym rozdziale pragnę przedstawić jedynie szkic algorytmu pozwalającego na symulację w czasie rzeczywistym. Jest to zadanie nieporównywalnie trudniejsze, aniżeli zbudowanie modelu „dokładnego”, jako, że wymaga uczynienia daleko idących uproszczeń. Dokonanie ich bez wnikliwej wiedzy na temat wpływu poszczególnych czynników na zachowanie mechanizmu jest praktycznie niemożliwe. Mówiąc o czynnikach mam tu na myśli przede wszystkim zjawiska tarcia, wymiany pędu pomiędzy elementami w trakcie ich zderzeń, tłumienie i właściwości sprężyste materiałów. Wiele z tych czynników można pominąć, inne zaś sformułować w taki sposób, aby wymagały minimum obliczeń.

Analizując mechanizm młoteczkowy doszedłem do wniosku, że zamiast traktować cały szereg występujących w nim dźwigni jako nieustannie „odbijające się” od siebie ciała można je zastąpić jednym ciałem *de facto* o jednym stopniu swobody. Jego elementy będą ze sobą niejako „posklejane” tak, że ich orientacja zależy będzie w sposób jednoznaczny od orientacji klawisza – narzuconej z góry. W modelu uproszczonym rolę wymuszenia spełniałaby nie siła nacisku na klawisz, ale jego uchylenie. Jedyne połączenie swobodne – nie sklejone - istniałoby pomiędzy figurą i młotkiem. Schematycznie obrazuje to rys. 31.:



Rys.31. Schemat modelu "dokładnego" i uproszczonego.

Najważniejsze uproszczenie dotyczy całkowania równań ruchu. Ponieważ zawierałyby one w swej jawniej postaci tylko momenty pochodzące od sprężyn i sił ciężkości, zatem wielkości bliskie stałym, wystarczyłoby najprostsze całkowanie metodą prostokątów :

$$\omega^{k+1} = \omega^k + \varepsilon^k dt$$

$$\alpha^{k+1} = \alpha^k + \omega^{k+1} dt \quad (32)$$

gdzie k oznacza numer liczonego kroku czasowego;

Schemat postępowania przykładowo dla figury jest następujący (proces nr 33):

1. wprowadzamy do układu jakieś wymuszenie w postaci drobnego uchylenia klawisza;
2. jeśli pilot dotyka stopkę, to obracamy figurę o pewien niewielki kąt (np. 0,001 rad w kierunku określonym dla każdego kontaktu w tablicy STY.zw) tyle razy, aż penetracja w kontakcie pilot-stopka będzie na poziomie nie więcej, niż np. 0.1mm; jednocześnie zerujemy prędkość figury („sklejenie”);
3. w przeciwnym razie przy ε^{k+1} i $\omega^{k+1} \neq 0$ obracamy figurę o kąt oszacowany na podstawie (32) – ruch swobodny;
4. znając ε^{k+1} figury równe $(\omega^{k+1} - \omega^k)/dt$ szacujemy siłę bezwładności wywieraną przez figurę na klawisz, a pośrednio siłę, jaką należałyby przyłożyć do klawisza, aby spowodować takie przyspieszenie figury;
5. wracamy do punktu 1.;

Analogicznie postępujemy z resztą ciał z tym, że :

- w przypadku bijnika i dźwigni pomijamy ich siły bezwładności, jako że są niewielkie; pomijamy również wpływ prędkości unoszenia przez figurę; uwzględniamy natomiast wszystkie możliwe kontakty pomiędzy bijnikiem, dźwignią, figurą, młotkiem i ciałem zerowym;
- w chwili styku młotka ze struną zamieniamy zwrot jego prędkości; w chwili styku z chwytnikiem zerujemy ją.
- jeśli prędkość młotka < 0 (obraca się w kierunku struny) bijnik lub dźwignia mogą jedynie rozpędzać; jeśli dochodzi między nimi do kontaktu, prędkość młotka zamiast być wyzerowaną jest stopniowo zwiększana proporcjonalnie do prędkości klawisza, aby przekazać młotkowi kręt - kiedy nastąpi wyzwolenie bijnika rozpędzony młotek ruchem swobodnym dotrze do struny;

Poniższy algorytm wykonuje wymienione kroki i w efekcie przy zadanym ugięciu klawisza zwraca on siłę, z jaką teoretycznie powinien nacisnąć palec na klawisz, aby ugiął się on o zadaną wartość. Jest to zatem zagadnienie odwrotne do postawionego w roz.3. Dopiero jednak tak sformułowane pozwalałoby zbudować instrument sterowany numerycznie, jako że znacznie łatwiej jest sterować siłą, aniżeli położeniem.

6. Model uproszczony.

```
AL0[6]:=AL[6];  
  
AL[6]:=(Mouse.y)*-11e-4;  
InOut;  
OM0[6]:=OM[6];  
OM[6]:=(AL[6]-AL0[6])/dt;  
EP[6]:=(OM[6]-OM0[6])/dt;  
  
for i:=1 to l_cial-1 do  
begin  
  AL0[i]:=AL[i];  
  OM0[i]:=OM[i];  
  ST[i].MO:=ST[i].Q*ST[i].S[1].w.x +MSp[i]-OM[i]*8e-5;  
  OM[i]:=OM[i]+ST[i].MO/ST[i].I*dt;  
end;  
  
for i:=1 to l_styk do  
begin  
  nk:=STY[i].kon;  
  y_kon(nk);  
  if(STY[i].str=1) then  
    begin  
      cg:=K[nk].c1;  
      pg:=K[nk].p11;  
    end;  
  if(STY[i].str=2) then  
    begin  
      cg:=K[nk].c2;  
      pg:=K[nk].p_r;  
    end;  
  
  if (KO[nk].y>0) then  
    begin  
      j:=0;  
      if(nk<>12) then  
        repeat  
          j:=j+1;  
          AL[cg]:=AL[cg]+ STY[i].zw * 1e-3;  
          Obl_pkt(1);  
          y_kon(nk);  
        until ((KO[nk].y<0.5e-3)or(j>100));  
      if(cg<>4) then  
        OM[cg]:=0;  
    end;  
  end;  
  if( ((KO[7].y>0)or(KO[6].y>0)) and (OM[4]>0) ) then  
    OM[4]:=0;  
  if((KO[6].y>0)and(k6_0>0)) then  
    begin  
      Omm:=11*OM[6];  
      if((Omm<0)and(Omm<OM[4])) then  
        OM[4]:=OM[4]+(Omm-OM[4])*0.7;  
      if(OM[4]<om_max) then  
        OM[4]:=om_max;  
    end;  
  if(KO[11].y>0) then  
    OM[4]:=abs(OM[4]);  
  if(KO[12].y>0) then  
    OM[4]:=0;
```

6. Model uproszczony.

```
for i:=1 to l_cial-1 do
begin
  EP[i]:=(AL[i]-AL0[i])-OM0[i];
  AL[i]:=AL[i]+OM[i]*dt;                                {ruch swobodny}
end;

ST[6].MO:=0;
if(KO[15].y>0) then
  ST[6].MO:=ST[6].MO+(-ST[5].Q*ST[5].S[1].w.x
                        +10*EP[5]*ST[5].I)/P[5,14].rb*P[6,7].rb;
if(KO[1].y>0) then
begin
  ST[6].MO:=ST[6].MO+(-(ST[1].Q+ST[2].Q+ST[3].Q)*ST[1].S[1].w.x
                        +10*EP[1]*ST[1].I)/P[1,5].rb*P[6,15].rb;
  if((KO[6].y>0)or(KO[7].y>0)) then
    ST[6].MO:=ST[6].MO+(+ST[4].Q*ST[4].S[1].w.x
                        -10*EP[4]*ST[4].I)
                        /P[4,19].rb*P[1,23].rb/P[1,5].rb*P[6,15].rb;
end;
if(ST[6].MO>0) then
  Reak:=ST[6].MO/0.21
else
  Reak:=0;
if(AL[6]<-0.01) then
  Reak:=Reak+2;
```

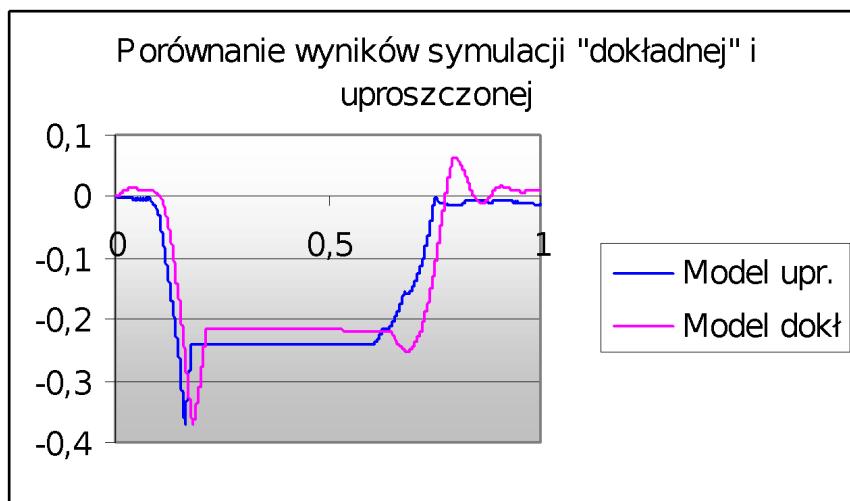
gdzie :

$$OM0 - \omega^k; AL0 - \alpha^k;$$

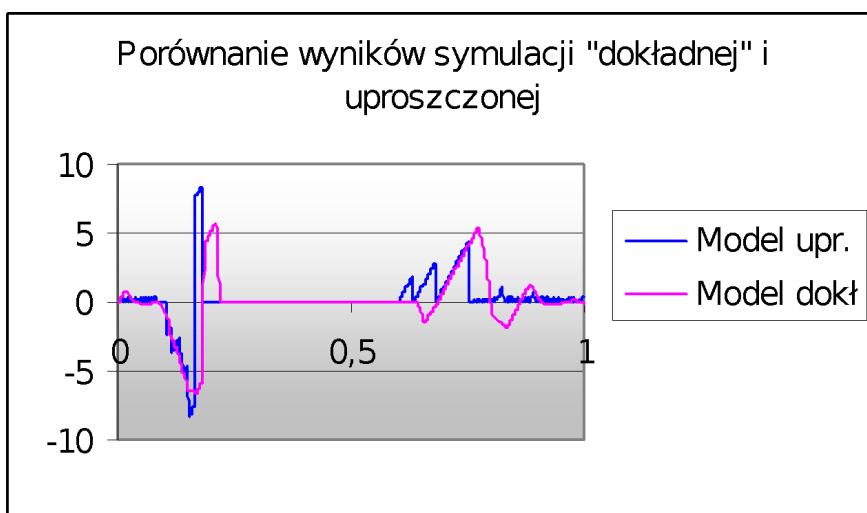
STY[].str - tablica zawierająca informację, które z ciał w kontakcie ma być obracane (np. dla kontaktu pilot-stopka obracana jest figura – *STY[]*.str=1; młotek-bijnik – *STY[]*.str=2);

Poniżej przedstawione jest porównanie wyników symulacji modelu "dokładnego" i uproszczonego. Są to :

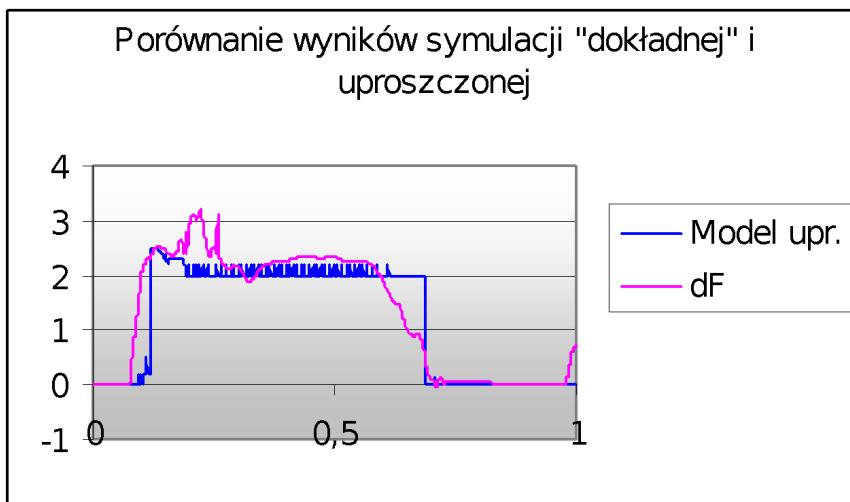
- położenia młoteczka;
- prędkości młoteczka;
- oszacowana siła, jaka wymusza ruch na klawisz, lub równoważy ciężar mechaniki (model uproszczony) w odniesieniu do siły wymuszenia dF zmierzanej doświadczalnie i pomniejszonej o siłę reakcji filcu dna klawiatury (obliczoną w symulacji modelu dokładnego);



Wyk.41. Porównanie położen młotka uzyskanych dla modelu "dokładnego" i uproszczonego.



Wyk.42. Porównanie prędkości młotka uzyskanych dla modelu "dokładnego" i uproszczonego.



Wyk.43. Porównanie sił wymuszających uzyskanych dla modelu "dokładnego" i uproszczonego.

Jak widać powyższe wyniki nie różnią się między sobą w sposób radykalny, a jest to przecież jedynie wstępna, niedopracowana wersja uproszczonego modelu. Wymaga ona jeszcze wielu ulepszeń, wielu miejscach ukazuje słabe strony. Jedno jest jednak pewne - symulacja oparta na takim modelu może działać w czasie rzeczywistym, co stanowi jej główną zaletę. Pojawia się naturalnie problem zadawania wymuszenia, a dokładnej częstotliwości odświeżania pozycji klawisza. Im jest ona większa, tym płynniej działa cały algorytm. Równocześnie jednak staje się on wolniejszy. Jeśli częstotliwość ta jest za mała, wówczas skokowe zmiany położenia klawisza mogą spowodować, że okręgi opisujące powierzchnie kontaktu dosłownie przejdą "na wylot" stykających się z nimi ciał opisanych odcinkami zanim punkt 2. procesu (33) zdąży skorygować ich położenia. Doprowadziłoby to do "załamania się" całej geometrii i uniemożliwiło dalszą symulację. Aby temu zapobiec konieczne jest odpowiednio częste odświeżanie pozycji klawisza.

7. Podsumowanie.

W niniejszej pracy pokazałem dwa podejścia do tego samego zagadnienia. Pierwsze - "dokładne" (używam znaku "", bo jest to jedynie przybliżenie rzeczywistego układu, pozostające jednak w zgodzie z obowiązującymi tu prawami mechaniki) pozwala przeanalizować zachowanie układu, ocenić wpływ poszczególnych czynników - przeprowadzić tzw. "Sensitivity Analysis".

Drugie podejście ma przeznaczenie bardziej praktyczne - pozwala na zbudowanie urządzenia symulującego prawdziwą klawiaturę fortepianu. Aby imitacja taka była dostatecznie wierna, niezbędna jest współpraca zarówno środowiska naukowego, jak i muzyków, którzy byliby w stanie ostatecznie zweryfikować model, w oparciu o który działałaby klawiatura. Na tym etapie badań rzeczą absolutnie niezbędną jest prototyp takiej klawiatury, pozwalający na ocenę "organoleptyczną" całego urządzenia.

Wszystko wskazuje więc na to, że od strony teoretycznej nie istnieją żadne poważne przeszkody uniemożliwiające realizację powyższego projektu. Zasadnicze problemy pojawiają się w związku z samym sterowaniem. Aby klawisz przenosił na palec pianisty opór klawiatury musi być wyposażony w sterowany przez komputer siłownik. Optymalnym wydaje się do tego celu siłownik elektromagnetyczny. Należy się jednak spodziewać licznych trudności w płynnym sterowaniu takim urządzeniem z uwagi na indukujące się w nim prądy poważnie zaburzające jego pracę. Są to jednak zagadnienia stanowczo wykraczające poza ramy niniejszej pracy.

Bibliografia :

- [1.] Baron B., *Metody Numeryczne w TP*, Helion, Gliwice 1995.
 - [2.] Fadiejew I., Ałłonow S., *Naprawa i strojenie pianin i fortepianów*, „Pomorze”, Bydgoszcz 1990.
 - [3.] Fletcher N., Rossing T., *The Physics of Musical Instruments*, New York, Springer-Verlag 1990.
 - [4.] Janczak M., *Biomechaniczne Aspekty Techniki Gry na Fortepianie*, Warszawa 2002.
 - [5.] Janczak M., *Eksperymentalne Wyznaczenie Parametrów Dynamicznych Wybranego Podzespołu Fortepianu*, Warszawa 2002.
 - [6.] Janczak M., *Symulacja Współczesnej Mechaniki Fortepianu przy Wykorzystaniu Programu MADYMO*, Warszawa 2002.
 - [7.] Jankowscy J. i M., *Przegląd Metod i Algorytmów Numerycznych*, WNT, Warszawa 1981.
 - [8.] Marciński A., Gregulec D., Kaczmarek J., *Podstawowe Procedury Numeryczne w Języku TP*, Wyd. Nakom, Poznań 1997.
 - [9.] Sielużycki Cz., *Ręka pianisty*, PWM, Warszawa, 1960.
 - [10.] Szmelter J., *Metody Komputerowe w Mechanice*, PWN, Warszawa 1980.
 - [11.] Tarnowski Z., *Modelowanie Matematyczne i Symulacja Komputerowa Dynamicznych Procesów Ciągłych*, Warszawa 1998.
 - [12.] Rabinowicz E., *The Nature of the Static and Kinetic Coefficients of Friction*, Jurnal of Applied Physic, Vol. 22, No 11, November, 1951
 - [13.] Waldorf J., *Ósme Sekrety Polihymnii*, Iskry, Warszawa 1997.
-

Unit Dane_4;

```
Interface
uses Dos,Crt,Graph,Objects,Drivers;
const
met_uk_row=3;
met_cal=2;

l_cial = 6;
l_pkt = 30;
l_kon = 17;
l_LM = 113;
l_kon_c = 5;
l_stop = 2;
n_mac = 3;
l_mat = 5;
l_p_mat = 14;
l_pop = 22;
l_help = 15;
l_etyk = 5;
l_out = 6;

File_pop= 'pop4.dat';
File_geo= 'geo4.dat';
File_map= 'map4.dat';
File_kon= 'kon4.dat';
File_mat= 'mat4.dat';
File_input='input_4.dat';
File_start='start.dat';
File_output='out4_0.dat';

pi = 3.1415926535897932385;
pip= pi/2;
r_s= 180/pi;
s_r= pi/180;
g = -9.81;
dt =0.001; {czestosc odswiezania wyk. i zapisu danych}
t_max=1; {czas zapisu}
mh =1e-7; {min krok całkowania}
eps=1e-6; {bl wzg całkowania}
eta=1e-6; {bl bezwzg całkowania}
max_it=10; {liczba iter.całkowania}

eps_g=1e-6; {bl wzg Gausa-Zaidla}
max_it_g=10;

eps_k=1e-5; {dop. błąd w obliczaniu kata}
sk_czasu = 3000;
sk_penet = 1e6;
sk_wymusz=0.1;

type
vec=array[1..l_cial] of Double;
matrix=array[1..l_cial,1..l_cial] of Double;
vector=array[1..2*l_cial] of Double;
fxysys=procedure (x : Double;
                  n : Integer;
                  y : vector;
```

```
    var f : vector);

vector_GJ = array[1..round((l_cial+2)*(l_cial+2)/4)] of Double;
vector1_GJ = array[1..l_cial+1] of Double;
vector2_GJ = array[1..l_cial+1] of integer;
coefficients = procedure (i,n : Integer;
                           var a : vector1_GJ);

pkt=record
  x,y : Double; end;
geom=record
  k,LM,pc,pp,r_ : integer;
  da,dr,dx,dy,ab,rb,a,r : Double;
  end;
kontakty_cons=record
  c1,p11,p12 : integer; z : Double;
  c2,p2,p_r : integer; rk : Double;
  tx:string[20];
  tar,wym : integer; mat : integer;
  ak,dk : Double; end;
_kontakty=record
  nK,p,zw : integer; w : pkt; a,r,m : Double; end;
_stopnie=record
  c,p : integer; w : pkt; a,r : Double; end;
hierarchia=record
  Mas,I : Double;
  tx : string[8];
  zak,Q : Double;
  sig: double;
  S : array[1..l_stop] of _stopnie;
  K : array[1..l_kon_c] of _kontakty;
  KO : pkt;
  M : array[1..l_cial] of integer;
  MO : Double;
  l_p,l_k,l_st : integer;
end;
_pr_wzg=record
  t,n,a,m : double; w : pkt; end;
kontakty_var=record
  x,N,T : Double;
  b,bT: array[1..2] of Double;
  Ft: array[1..3] of Double;
  vw : _pr_wzg;
  w,wT : array[1..2] of pkt; end;
Ekran=record
  x,y : integer; s : Double; end;
_mat=record
  x,yob,aob,yod,aod : double; end;
material=record
  naz: string[8]; v0_N,m_s,m_d,v0_T,dF: Double;
  T : array[0..l_p_mat] of _mat; end;
sprez=record
  pc : integer; m0,m,dm0,dm : Double; end;
time=record
  h,m,s,ms : word; end;
time_=record
  h,m,s,ms : Longint ; end;
okna = record
  x1,y1,x2,y2 : integer; end;
wykres = record
  y0,xx,yy : integer;
```

```
x,y,s : Double; t,j : string[25]; end;
etykiety = record
  x,s : Double; t : string[5]; end;
poprawki = record
  tyt : string[25]; ob,po : string[2];
  c,p : integer; j,pop : Double; end;
out_data = record
  tx : string[8]; _0,dat : double; end;
start_data = record
  t : double; Y : vector; end;

const
Win : array[0..8] of okna=
((x1:0;    y1:0;    x2:640; y2:350),
 (x1:4;    y1:13;   x2:318; y2:93),
 (x1:4;    y1:97;   x2:318; y2:177),
 (x1:322;  y1:13;   x2:636; y2:93),
 (x1:322;  y1:97;   x2:636; y2:177),
 (x1:4;    y1:181;  x2:636; y2:346),
 (x1:4;    y1:13;   x2:318; y2:177),
 (x1:322;  y1:13;   x2:636; y2:177),
 (x1:4;    y1:13;   x2:636; y2:177));

ST0    : array[0..1_cial] of hierarchia=
(),,
(Mas:0.011; I:3.4e-5; tx:'figura'; zak:15),
(Mas:5.8e-3; I:1.0e-5; tx:'dzw.rep';zak:-15),
(Mas:3.2e-3; I:2.5e-6; tx:'bijnik'; zak:-15),
(Mas:9.2e-3; I:8.9e-5; tx:'młotek'; zak:-50),
(Mas:3.1e-2; I:3.5e-4; tx:'tłumik'; zak:20),
(Mas:0.112; I:5.4e-3; tx:'klawisz';zak:-4 ));

Tlum   : array[1..1_cial] of double=
((8e-5),(8e-5),(8e-5),(1e-4),(1e-2),(1e-2));

SP0    : array[0..1_cial] of sprez= {pc:-1 - wyłączenie sprężyny}
(),,
(pc:-1),
(pc:1; m0:0.05; m:-0.01),
(pc:1; m0:0.005; m:-0.005),
(pc:-1),
(pc:-1),
(pc:-1));

p00    : pkt=(x:0; y:0);

Help : array[1..1_help] of string=
('s - Start'),
('q - Koniec'),
('p - Pauza'),
('F1 - Help'),
('x,t - Zmiana wykresów'),
('o - Odświeżanie wykresów'),
('e - Etykiety przy ciałach lub kontaktach'),
('PgUp/PgDn - Zmiana pokazywanego kontaktu '),
('Home/End - Zmiana pokazywanego ciała '),
('+/-;4,6,8,2 - Zoom; Przewijanie'),
('d - Poprawki dr, da, dx, dy'),
('    F2 - Zapis poprawek ('+File_pop+')'),
('    F3 - Odczyt poprawek ('+File_pop+')',
('    ^ , v - Wybór parametru'),
```

```

('      < , > - Korekta parametru'));

var  {tab. daych}
P
K
SP
POP
MAT
{zmienne}
PO,VE,AC
et_k,et_c
KO
EP,MSp
EP_
AL,OM
ST
OUT
AA
BB
Y
tY_start
t
t_
Wyk
VPort
Ekr
GraphDriver,GraphMode
st_cal,st_gau,it_gau,Vpg
n_pop,i_kon,n_kon,n_cial
t0,t1,t_r,t_wyk,t_in
h,Wymusz
f1,czysc_,pauza,start
et1,et2,et3,et4,et5
s1,s2,s3
klaw,f1,ff1,f2,ff2
Event
Mouse
Fpop
Fmap
Fgeo
Fmat
Fkon
Fsta
F_in,Fout
function sign(x : Double) : integer;
function nor_w(w : vec) : Double;
function nor_k(kat : Double) : Double;
function dl_wek(P1,P2 : pkt) : Double;
function k_wek(P1,P2 : pkt) : Double;
function il_wek(P1,P2 : pkt) : Double;
function il_sk(W1,W2 : pkt) : Double;
procedure popraw;

```

Implementation

```

function sign(x : Double) : integer;
begin
  if (x<0) then sign:=-1
  else sign:=1;
end;

```

```
function nor_w(w : vec) : Double;
var i : integer;
m : Double;
begin
  m:=abs(w[1]);
  for i:=2 to l_cial do
    if (abs(w[i])>m) then
      m:=abs(w[i]);
  nor_w:=m;
end;

function nor_k(kat : Double) : Double;
var st_ : integer;
begin
repeat
  st_:=0;
  if (kat>pi) then
    begin
      st_:=1;
      kat:=kat-2*pi;
    end;
  if (kat<=-pi) then
    begin
      st_:=1;
      kat:=kat+2*pi;
    end;
until(st_=0);
nor_k:=kat;
end;

function dl_wek(P1,P2 : pkt) : Double;
begin
  dl_wek:=sqrt(sqr(P2.x-P1.x)+sqr(P2.y-P1.y));
end;

function k_wek(P1,P2 : pkt) : Double;
var kat,dx,dy : Double;
begin
  dx:=P2.x-P1.x;
  dy:=P2.y-P1.y;
  if(abs(dx)>eps_k) then
    kat:=arctan(dy/abs(dx))
  else kat:=sign(dy)*pi;
  if(dx<0) then
    kat:=sign(kat)*pi-kat;
  kat:=nor_k(kat);
  k_wek:=kat;
end;

function il_sk(W1,W2 : pkt) : Double;
begin
  il_sk:= W1.x*W2.x + W1.y*W2.y;
end;

function il_wek(P1,P2 : pkt) : Double;
begin
  il_wek:=P1.x*P2.y-P1.y*P2.x;
end;
```

```
procedure popraw;
var i :integer;
begin
  for i:=1 to l_pop do
  begin
    if(POP[i].ob='P ') then
    begin
      if(POP[i].po='a ') then
        P[POP[i].c,POP[i].p].da:=POP[i].pop;
      if(POP[i].po='r ') then
        P[POP[i].c,POP[i].p].dr:=POP[i].pop;
      if(POP[i].po='x ') then
        P[POP[i].c,POP[i].p].dx:=POP[i].pop;
      if(POP[i].po='y ') then
        P[POP[i].c,POP[i].p].dy:=POP[i].pop;

      if((POP[i].c=3)and(POP[i].p=4)and(POP[i].po='r ')) then
        P[3,6].dr:=P[3,4].dr; {dł. bijnika}

    end;
    if(POP[i].ob='SP') then
    begin
      if(POP[i].po='m0') then
        SP[POP[i].c].dm0:=POP[i].pop;
      if(POP[i].po='m ') then
        SP[POP[i].c].dm:=POP[i].pop;
    end;
  end;
end;
```

end.

Unit Grafika4;

```
Interface
Uses Dos,Crt,Graph,Objects,Drivers,Dane_4;

procedure ViewPorty(n : integer);
procedure Schemat(pp : char);
procedure Wykresy(n0,n1 : integer);
procedure Poprawki(n0 : integer);
procedure Help_(n0 : integer);
procedure Ekran(n0,n1 : integer);
procedure Wpisz_dane;
procedure Rysuj(f1,f2 : char);

Implementation

procedure ViewPorty(n : integer);
begin
  setViewPort(Win[n].x1,Win[n].y1,Win[n].x2,Win[n].y2,true);
end;

procedure Schemat(pp : char);
var i,j,x,zw : integer;
  alf,wsp:double;
  l : okna;
begin
  ViewPorty(5);
  ClearViewPort;
  for i:=0 to l_cial do
    begin
      if (ST[i].l_st=1) then          {symbole osi główne}
        begin
          setcolor(5);
          l.x1:=Ekr.x+round(Ekr.s*PO[i,1].x);
          l.y1:=Ekr.y-round(Ekr.s*PO[i,1].y);
          line(l.x1-5,l.y1+10,l.x1+5,l.y1+10);
          line(l.x1,l.y1,l.x1+5,l.y1+10);
          line(l.x1,l.y1,l.x1-5,l.y1+10);
        end;
      if (ST[i].l_st>1) then          {symbole osi II stopnia}
        begin
          setcolor(5);
          l.x1:=Ekr.x+round(Ekr.s*PO[i,1].x);
          l.y1:=Ekr.y-round(Ekr.s*PO[i,1].y);
          line(l.x1-2,l.y1+4,l.x1+2,l.y1+4);
          line(l.x1,l.y1,l.x1+2,l.y1+4);
          line(l.x1,l.y1,l.x1-2,l.y1+4);
        end;
      for j:=1 to ST[i].l_p do
        if (P[i,j].r_=1) then          {kontury}
          begin
            setcolor(3);
            l.x1:=Ekr.x+round(Ekr.s*PO[P[i,j].pc,P[i,j].pp].x);
            l.y1:=Ekr.y-round(Ekr.s*PO[P[i,j].pc,P[i,j].pp].y);
            l.x2:=Ekr.x+round(Ekr.s*PO[i,j].x);
            l.y2:=Ekr.y-round(Ekr.s*PO[i,j].y);
            line(l.x1,l.y1,l.x2,l.y2);
          end;
    end;
end;
```



```

begin
  setcolor(white);
  l.x1:=Ekr.x+round(Ekr.s*PO[K[n_kon].c2,K[n_kon].p_r].x);
  l.y1:=Ekr.y-round(Ekr.s*PO[K[n_kon].c2,K[n_kon].p_r].y);
  line(l.x1,l.y1,l.x1+2,l.y1-5);
  line(l.x1,l.y1,l.x1+5,l.y1-2);
  line(l.x1,l.y1,l.x1+10,l.y1-10);
  if(KO[n_kon].N>0) then
    setcolor(red)
  else
    setcolor(green);
  outtextxy(l.x1+20,l.y1-10,K[n_kon].tx);
  outtextxy(500,70,K[n_kon].tx);
  setcolor(3);
  outtextxy(500,60,ST[n_cial].tx);
                                {legenda}
  setlinestyle(1,1,1);
  setcolor(62);   line(500,135,550,135);
                  outtextxy(550,130,'V kon');
  setcolor(58);   line(500,135+8,550,135+8);
                  outtextxy(550,130+8,'F wym');
  setlinestyle(0,1,1);
  setcolor(57);   line(500,135+16,550,135+16);
                  outtextxy(550,130+16,'N kon');
  setcolor(60);   line(500,135+24,550,135+24);
                  outtextxy(550,130+24,'T kon');

  zw:=ST[n_cial].K[i_kon].zw;
  case zw of
    1 : begin
      l.x1:=Ekr.x+round(Ekr.s*PO[ K[n_kon].c1,
                                     K[n_kon].p11 ].x);
      l.y1:=Ekr.y-round(Ekr.s*PO[ K[n_kon].c1,
                                     K[n_kon].p11 ].y);
    end;
    2 : begin
      l.x1:=Ekr.x+round(Ekr.s*PO[ K[n_kon].c2,
                                     K[n_kon].p_r ].x);
      l.y1:=Ekr.y-round(Ekr.s*PO[ K[n_kon].c2,
                                     K[n_kon].p_r ].y);
    end;
  end;
  setlinestyle(0,1,1);
  l.x2:=l.x1+round(50*KO[n_kon].w[zw].x);
  l.y2:=l.y1-round(50*KO[n_kon].w[zw].y);
  setcolor(57);
  line(l.x1,l.y1,l.x2,l.y2);
  l.x2:=l.x1+round(50*KO[n_kon].wT[zw].x);
  l.y2:=l.y1-round(50*KO[n_kon].wT[zw].y);
  setcolor(60);
  line(l.x1,l.y1,l.x2,l.y2);

  setlinestyle(1,1,1);
  l.x2:=l.x1+round(50*cos(KO[n_kon].b[zw]-pip)*KO[n_kon].Ft[3]);
  l.y2:=l.y1-round(50*sin(KO[n_kon].b[zw]-pip)*KO[n_kon].Ft[3]);
  setcolor(58);
  line(l.x1,l.y1-1,l.x2,l.y2);
  if (ST[n_cial].K[i_kon].zw=1) then
    wsp:=1.0
  else
    wsp:=-1.0;

```



```
procedure Poprawki(n0 : integer);
var i,j,x,y : integer;
begin
    GraphDefaults;
    ViewPorty(n0);
    ClearViewPort;
    case klaw of
        #72 : begin
            n_pop:=n_pop-1;
            if(n_pop<=2) then
                n_pop:=l_pop;
            klaw:=' ';
        end;
        #80 : begin
            n_pop:=n_pop+1;
            if(n_pop>l_pop) then
                n_pop:=3;
            klaw:=' ';
        end;
        #75 : begin
            POP[n_pop].pop:=POP[n_pop].pop-POP[n_pop].j;
            klaw:=' ';
        end;
        #77 : begin
            POP[n_pop].pop:=POP[n_pop].pop+POP[n_pop].j;
            klaw:=' ';
        end;
        #60 : begin
            Assign(Fpop,File_pop);
            Rewrite(Fpop);
            SetActivePage(1);
            SetVisualPage(1);
            for i:=1 to l_pop do
                Write(Fpop,POP[i]);
            if(filesize(Fpop)=l_pop) then
                OutTextXY(20,130,'Plik zapisany pomyślnie')
            else
                OutTextXY(20,130,'Plik niezapisany!');
            delay(1000);
            close(Fpop);
            klaw:=' ';
        end;
        #61 : begin
            Assign(Fpop,File_pop);
            Reset(Fpop);
            SetActivePage(1);
            SetVisualPage(1);
            if ((IOResult=0)and(filesize(Fpop)=l_pop)) then
                begin
                    for i:=1 to l_pop do
                        Read(Fpop,POP[i]);
                    OutTextXY(20,130,'Plik załadowany pomyślnie')
                end
            else
                OutTextXY(20,130,'Plik niezaładowany!');
            delay(1000);
            close(Fpop);
            klaw:=' ';
        end;
    end;
end;
```

```
for i:=0 to 1 do
begin
  SetActivePage(i);
  for j:=1 to l_pop do
    begin
      if(j=n_pop) then
        setcolor(white)
      else
        setcolor(7);
      str(POP[j].c:2,s1);
      str(POP[j].p:2,s2);
      str(POP[j].pop:6:4,s3);
      if(POP[j].ob=' ') then
        begin s1:=' ';s2:=' ';s3:=' ';end;
      if(j<=12) then begin x:=0; y:=0; end
      else begin x:=300; y:=100; end;
      OutTextXY(20+x,j*10-y,POP[j].tyt+' '+POP[j].ob+
'+POP[j].po+' '+s1+' '+s2+' '+s3);
    end;
  OutTextXY(20,140,'F2 - Zapis; F3 - Odczyt;');
end;

procedure Help_(n0 : integer);
var i,j : integer;
begin
  GraphDefaults;
  ViewPorty(n0);
  setcolor(white);
  for i:=0 to 1 do
    begin
      SetActivePage(i);
      for j:=1 to l_help do
        OutTextXY(20,j*10,help[j]);
    end;
end;

procedure Ekran(n0,n1 : integer);
var i : integer;
begin
  SetColor(yellow);
  SetFillStyle(1,14);
  Bar(1,1,639,10);
  SetLineStyle(0,1,1);
  Rectangle(1,1,639,349);
  SetColor(blue);
  OutTextXY(10,2,'Symulacja Mechaniki Fortepianu ver.4.0');
  for i:=n0 to n1 do
    begin
      SetColor(yellow);
      SetLineStyle(0,1,1);
      Rectangle(Win[i].x1-1,Win[i].y1-1,Win[i].x2+1,Win[i].y2+1);
      SetLineStyle(3,1,1);
      if(i<8) then
        begin
          SetColor(7);
          line(Win[i].x1,Wyk[i].y0+Win[i].y1,Win[i].x2,Wyk[i].y0+Win[i].y1);
          str(70/Wyk[i].s:5:3,s1);
          s2:=s1+Wyk[i].j;
          SetColor(2);
          OutTextXY(Win[i].x1+1,Win[i].y1+1,s2);
        end;
    end;
end;
```

```

        OutTextXY(Win[i].x1+100,Win[i].y1+1,Wyk[i].t);
    end;
    if(f2='t') then begin
        str(t_wyk:5:3,s1);
        str(t_wyk+314/sk_czasu:5:3,s2);
        s3:='Czas '+s1+' - '+s2+' s';
        SetColor(2);
        OutTextXY(Win[i].x1+30,Win[i].y2-8,s3); end;
    if(f2='x') then begin
        str(314/sk_penet*1000:5:3,s1);
        s2:='Penetracja 0 - '+s1+' mm';
        SetColor(2);
        OutTextXY(Win[i].x1+30,Win[i].y2-8,s2); end;
    end;
    SetColor(yellow);
    SetLineStyle(0,1,1);
    Rectangle(Win[5].x1-1,Win[5].y1-1,Win[5].x2+1,Win[5].y2+1);
end;

procedure Wpisz_dane;
var i :integer;
begin
    OUT[1].dat:=t0;           OUT[1].tx:='czas      ';
    OUT[2].dat:=AL[6];       OUT[2].tx:='alf klaw';
    OUT[3].dat:=AL[4];       OUT[3].tx:='alf mlot';
    OUT[4].dat:=OM[6];       OUT[4].tx:='om klaw ';
    OUT[5].dat:=OM[4];       OUT[5].tx:='om mlot ';
    OUT[6].dat:=KO[17].N;    OUT[5].tx:='reak.dna';
end;

procedure Rysuj(f1,f2 : char);
var v1,v2,alf,bet : double;
n0,n1 : integer;
var i,j,zw : integer;
czysc : boolean;
begin
    zw:=ST[n_cial].K[i_kon].zw;
    et_k[1].x:=KO[n_kon].N;          et_k[1].t:='N   ';
    et_k[2].x:=KO[n_kon].vw.n;       et_k[2].t:='vn  ';
    et_k[3].x:=KO[n_kon].vw.t;       et_k[3].t:='vt  ';
    et_k[4].x:=KO[n_kon].vw.a*r_s;  et_k[4].t:='v_a ';
    et_k[5].x:=KO[n_kon].x*1000;    et_k[5].t:='x[mm] ';
    et_c[1].x:=MSP[n_cial];         et_c[1].t:='MSP';
    et_c[2].x:=ST[n_cial].MO;       et_c[2].t:='MO ';
    et_c[3].x:=AL[n_cial];         et_c[3].t:='AL ';
    et_c[4].x:=OM[n_cial];         et_c[4].t:='OM ';
    et_c[5].x:=EP[n_cial];         et_c[5].t:='EP ';

    czysc:=false;
    if(czysc_) then
        if((t0-t_wyk)*sk_czasu>314) then
            begin
                t_wyk:=t0;
                for i:=1 to 4 do
                    Wyk[i].xx:=0;
                czysc:=true;
            end;
        for i:=1 to 4 do
            Wyk[i].x:=(t0-t_wyk)*sk_czasu;
            if(Wyk[i].x>1000) then
                Wyk[i].x:=1000;
    case f1 of

```

```

'c' : begin
    Wyk[1].y:=Wymusz;      Wyk[1].s:=12;
    Wyk[1].t:='Wymuszenie';      Wyk[1].j:='N';
    j:=11; Wyk[2].y:=KO[j].N;      Wyk[2].s:=30;
    Wyk[2].t:='Nacisk '+K[j].tx;      Wyk[2].j:='N';
    j:=6;  Wyk[3].y:=AL[j]*r_s;      Wyk[3].s:=70/ST[j].zak;
    Wyk[3].t:='Położ. '+ST[j].tx;      Wyk[3].j:='st';
    j:=4;  Wyk[4].y:=AL[j]*r_s;      Wyk[4].s:=70/ST[j].zak;
    Wyk[4].t:='Położ. '+ST[j].tx;      Wyk[4].j:='st';
    end;
    'k', ' ' : begin
    Wyk[1].y:=KO[n_kon].T;      Wyk[1].s:=30;
    Wyk[1].t:='T '+K[n_kon].tx; Wyk[1].j:='N';
    Wyk[2].y:=et2;      Wyk[2].s:=30;
    Wyk[2].t:='T_s '+K[n_kon].tx; Wyk[2].j:='N';
    Wyk[3].y:=et3;      Wyk[3].s:=30;
    Wyk[3].t:='T_d '+K[n_kon].tx; Wyk[3].j:='N';
    Wyk[4].y:=et5;      Wyk[4].s:=50;
    Wyk[4].t:='w_d '+K[n_kon].tx; Wyk[4].j:=' ';
    end;
    end;
    j:=1;  Wyk[6].x:=KO[j].x*sk_penet;
    Wyk[6].y:=KO[j].N; Wyk[6].s:=10;
    Wyk[6].t:='Histereza '++K[j].tx; Wyk[6].j:='N';
    Wyk[7].x:=KO[n_kon].x*sk_penet;
    Wyk[7].y:=KO[n_kon].N; Wyk[7].s:=50;
    Wyk[7].t:='Histereza '+K[n_kon].tx; Wyk[7].j:='N';
    case f2 of
    't' : begin n0:=1; n1:=4; end;
    'x' : begin n0:=6; n1:=7; end;
    'd',#59 : begin n0:=8; n1:=8; end;
    else begin n0:=8; n1:=8; end;
    end;
    if ((start)or(f2<>ff2)or(f1<>ff1)or((f2='t')and(czysc))) then
        for i:=0 to 1 do
            begin
                GraphDefaults;
                SetActivePage(i);
                ViewPort(0);
                ClearViewPort;
                Ekran(n0,n1);
                ff2:=f2;
                ff1:=f1;
                start:=false;
                for j:=6 to 7 do
                    Wyk[j].x:=0;
                end;
                SetActivePage(1-Vpg);
                GraphDefaults;
                Schemat(f1);
                GraphDefaults;
                case f2 of
                't','x' : Wykresy(n0,n1);
                'd' : Poprawki(n0);
                #59 : Help_(n0);
                end;
                Vpg:=1-Vpg;
                SetVisualPage(Vpg);
            end;
        end.
    
```

Unit Pr_num4;

```
Interface
Uses Dane_4,crt;

function max(n : integer; w : vector) : Double;
function min(n : integer; w : vector) : Double;
procedure krok(x0 : Double;
               hh : Double;
               y0 : vector;
               var y1,E : vector;
               n : integer;
               fun      : fxysys);

procedure Fehlberg1(var x0      : Double;
                     x1      : Double;
                     n       : Integer;
                     var y    : vector;
                     fun     : fxysys;
                     mh,eps,eta : Double;
                     m_it    : Integer;
                     var stan : Integer;
                     var h    : Double);

procedure GaussSeidel (n      : Integer;
                       var a    : matrix;
                       var b    : vec;
                       mit    : Integer;
                       eps   : Double;
                       var x   : vec;
                       var it,st : Integer);

procedure Wyp_mac(i,n : integer;
                  var a : vector1_GJ);
```

Implementation

```
function max(n : integer; w : vector) : Double;
var i : integer;
m : Double;
begin
  m:=abs(w[1]);
  for i:=2 to n do
    if (abs(w[i])>m) then
      m:=abs(w[i]);
  max:=m;
end;

function min(n : integer; w : vector) : Double;
var i : integer;
m : Double;
begin
  m:=abs(w[1]);
  for i:=2 to n do
    if (abs(w[i])<m) then
```

```

m:=abs(w[i]);
min:=m;
end;

procedure Krok(x0 : Double;
               hh : Double;
               y0 : vector;
               var y1,E : vector;
               n : integer;
               fun           : fxysys);
const
c : array[1..6] of real=
(0,1/4,3/8,12/13,1,1/2);
a : array[1..6,1..5] of real=
((0,0,0,0,0),
 (1/4,0,0,0,0),
 (3/32,      9/32,      0,      0,      0),
 (1932/2197,-7200/2197, 7296/2197, 0,      0),
 (439/216,   -8,        3680/513, -845/4104, 0),
 (-8/27,     2,        -3544/2565, 1859/4104,-11/40));
wE : array[1..6] of real=
(1/360,0,-128/4275,-2197/75240,1/50,2/55);
wY : array[1..6] of real=
(16/135,0,6656/12825,28561/56437,-9/50,2/55);

var i,j,l : integer;
yy,ff : vector;
K : array[1..6] of vector;
begin
    fun(x0,n,y0,ff);
    for i:=1 to n do K[1,i]:=ff[i]*hh;
    for l:=2 to 6 do
    begin
        for i:=1 to n do
        begin
            yy[i]:=y0[i];
            for j:=1 to l-1 do
                yy[i]:=yy[i]+a[l,j]*K[l-1,i];
        end;
        fun(x0+c[l]*hh,n,yy,ff);
        for i:=1 to n do K[1,i]:=ff[i]*hh;
    end;

    for i:=1 to n do
    begin
        y1[i]:=y0[i];
        E[i]:=0;
        for j:=1 to 6 do
        begin
            y1[i]:=y1[i]+wY[j]*K[j,i];
            E[i]:=E[i]+wE[j]*K[j,i];
        end;
    end;
end;

procedure Fehlberg1(var x0      : Double;
                     x1      : Double;
                     n       : Integer;
                     var y    : vector;
                     fun     : fxysys;
                     mh,eps,eta : Double;

```

```

        m_it      : Integer;
        var stan   : Integer;
        var h      : Double);

var i,j,l,it : integer;
y0,y1,f,E : vector;
dop_b1,b1_,alf : Double;
begin
  for i:=1 to n do y0[i]:=y[i];
  if(x1>x0) then
  repeat
    it:=0;
    repeat
      stan:=-1;
      krok(x0,h,y0,y1,E,n,fun);
      dop_b1:=eps*max(n,y1)+eta;
      b1_:=max(n,E);
      alf:=exp(-1/6*ln(b1_/dop_b1))*0.9;
      if (alf>5) then alf:=5;
      h:=alf*h;
      it:=it+1;
      if (b1_>dop_b1) then stan:=-1
      else stan:=0;
      if (h<mh) then stan:=1;
      if (it>max_it) then stan:=2;
      if (stan>0) then begin
        sound(1000);
        delay(100);
        nosound; end;
      until (stan>=0);
      x0:=x0+h;
      for i:=1 to n do y0[i]:=y1[i];

      until ((x0>=x1)or(stan>0));
      for i:=1 to n do y[i]:=y1[i];
end;

procedure GaussSeidel (n      : Integer;
                       var a      : matrix;
                       var b      : vec;
                       mit     : Integer;
                       eps     : Double;
                       var x      : vec;
                       var it,st : Integer);
{-----
{
{ The procedure GaussSeidel solves a system of linear equations by the
{ Gauss-Seidel iterative method.
{ Data:
{   n   - number of equations = number of unknowns,
{   a   - a two-dimensional array containing elements of the matrix of the
{         system (changed on exit),
{   b   - a one-dimensional array containing free terms of the system
{         (changed on exit),
{   mit - maximum number of iterations in the Gauss-Seidel method,
{   eps - relative accuracy of the solution,
{   x   - an array containing an initial approximation to the solution
{         (changed on exit).
{ Results:
{   x   - an array containing the solution,
{   it - number of iterations.
}

```

```

{ Other parameters: } }
{ st - a variable which within the procedure GaussSeidel is assigned the } }
{ value of: } }
{   1, if n<1, } }
{   2, if the matrix of the system is singular, } }
{   3, if the desired accuracy of the solution is not achieved in } }
{       mit iteration steps, } }
{   0, otherwise. } }

{ Note: If st=1 or st=2, then the elements of array x are not } }
{       changed on exit. If st=3, then x contains the last } }
{       approximation to the solution. } }

{Unlocal identifiers: } }
{   vector - a type identifier of Double array [q1..qn], where q1<=1 and } }
{       qn>=n, } }
{   matrix - a type identifier of Double array [q1..qn,q1..qn], where } }
{       q1<=1 and qn>=n. } }
{-----} }

var i,ih,k,khh,lz1,lz2 : Integer;
    max,r : Double;
    cond : Boolean;
    x1 : vector;

begin
  if n<1
    then st:=1
    else begin
      st:=0;
      cond:=true;
      for k:=1 to n do
        x1[k]:=0;
      repeat
        lz1:=0;
        khh:=0;
        for k:=1 to n do
          begin
            lz2:=0;
            if a[k,k]=0
              then begin
                kh:=k;
                for i:=1 to n do
                  if a[i,k]=0
                    then lz2:=lz2+1;
                if lz2>lz1
                  then begin
                    lz1:=lz2;
                    khh:=kh
                  end
                end
              end;
            if khh=0
              then cond:=false
            else begin
              max:=0;
              for i:=1 to n do
                begin
                  r:=abs(a[i,khh]);
                  if (r>max) and (x1[i]=0)
                    then begin
                      max:=r;
                      ih:=i
                    end
                end
              end
            end;
          end;
        end;
      end;
    end;
  if cond
    then writeln('The solution is not found')
    else begin
      writeln('The solution is found');
      for i:=1 to n do
        writeln('x',i,' = ',x1[i]);
    end;
  end;
end.

```

```

        end;
        if max=0
        then st:=2
        else begin
            for k:=1 to n do
            begin
                r:=a[khh,k];
                a[khh,k]:=a[ih,k];
                a[ih,k]:=r
            end;
            r:=b[khh];
            b[khh]:=b[ih];
            b[ih]:=r;
            x1[khh]:=1
            end
        end
    until not cond or (st=2);
    if not cond
    then begin
        it:=0;
        repeat
            it:=it+1;
            if it>mit
            then begin
                st:=3;
                it:=it-1
            end
            else begin
                for i:=1 to n do
                begin
                    r:=b[i];
                    for k:=1 to i-1 do
                        r:=r-a[i,k]*x[k];
                    for k:=i+1 to n do
                        r:=r-a[i,k]*x1[k];
                    x1[i]:=r/a[i,i]
                end;
                cond:=true;
                i:=0;
                repeat
                    i:=i+1;
                    max:=abs(x[i]);
                    r:=abs(x1[i]);
                    if max<r
                    then max:=r;
                    if max<>0
                    then if abs(x[i]-x1[i])/max>=eps
                        then cond:=false
                until (i=n) or not cond;
                for i:=1 to n do
                    x[i]:=x1[i]
                end
            until (st=3) or cond
        end
    end;
end;

procedure Wyp_mac(i,n : integer;
                  var a : vector1_GJ);
var j : integer;
begin

```

```
for j:=1 to n_mac do
  a[j]:=AA[i,j];
  a[n_mac]:=BB[i];
end;

end.
```

Program Piano4;

```
program piano_4;
{$N+}
uses Dos,Crt,Graph,Objects,Drivers,Dane_4,Pr_num4,Grafika4;
{-----}
procedure Obl_pkt(zak : integer);
var i,j,k,pp,cc : integer;
alf,r:Double;
pPO : pkt;
begin
for i:=1 to l_cial do
  for j:=1 to ST[i].l_p do
    begin
      if (P[i,j].k>=zak) then
        begin
          pPO:=PO[P[i,1].pc,P[i,1].pp];
          if(j=1) then
            PO[i,j]:=pPO
          else
            begin
              PO[i,j].x:=pPO.x+cos(P[i,j].ab+AL[i])*P[i,j].rb;
              PO[i,j].y:=pPO.y+sin(P[i,j].ab+AL[i])*P[i,j].rb;
            end;
        end;

      if (P[i,j].k=1) then
        begin
          alf:=P[i,j].ab+AL[i];
          r:=P[i,j].rb;
          VE[i,j].x:=-sin(alf)*r*OM[i];
          VE[i,j].y:= cos(alf)*r*OM[i];
          for k:=2 to ST[i].l_st do
            begin
              alf:=ST[i].S[k].a+AL[ST[i].S[k].c];
              r:=ST[i].S[k].r;
              cc:=ST[i].S[k].c;
              VE[i,j].x:=VE[i,j].x-sin(alf)*r*OM[cc];
              VE[i,j].y:=VE[i,j].y+cos(alf)*r*OM[cc];
            end;
        end;
    end;

  for i:=1 to l_cial do
    for j:=1 to ST[i].l_st do
      begin
        ST[i].S[j].w.x:=ST[i].S[j].r*cos(ST[i].S[j].a+AL[ST[i].S[j].c]);
        ST[i].S[j].w.y:=ST[i].S[j].r*sin(ST[i].S[j].a+AL[ST[i].S[j].c]);
      end;
  for i:=1 to l_cial do
    for j:=1 to ST[i].l_k do
      begin
        ST[i].K[j].w.x:=ST[i].K[j].r*cos(ST[i].K[j].a+AL[i]);
        ST[i].K[j].w.y:=ST[i].K[j].r*sin(ST[i].K[j].a+AL[i]);
      end;
end;
{-----}
function R_mat(m : integer; x,v : Double) : Double;
```

```

var w,Fob,Fod : Extended;
i,j : integer;
ex : boolean;
begin
  if(v<Mat[m].v0_N) then
    w:=0;
  if((v>=Mat[m].v0_N) and (v<0)) then
    w:=(v-Mat[m].v0_N)/Mat[m].v0_N;
  if(v>=0) then
    w:=1;
  i:=0; ex:=false;
repeat
  i:=i+1;
  if(i>l_p_mat) then
    begin
      i:=l_p_mat;
      x:=Mat[m].T[i].x
    end;
  if(x<=Mat[m].T[i].x) then
    begin
      Fob:=(x-Mat[m].T[i-1].x)*Mat[m].T[i].aob+Mat[m].T[i-1].yob;
      Fod:=(x-Mat[m].T[i-1].x)*Mat[m].T[i].aod+Mat[m].T[i-1].yod;
      ex:=true;
    end;
  until (ex);

R_mat:=(w*Fob+(1-w)*Fod);

end;
{-----}
procedure Mom_spr;
var i : integer;
begin
  for i:=1 to l_cial do
    begin
      if Sp[i].pc>=0 then
        MSp[i]:=Sp[i].m0+Sp[i].m*nor_k(AL[i]-AL[Sp[i].pc])
      else MSp[i]:=0;

      MSp[i]:=MSp[i]-OM[i]*tlum[i];
    end;
end;
{-----}
procedure Sil_kon;
var i,j,st_k,nK,zw : integer;
xx,x,d,F,alf,bet,gam : Double;
begin
  for i:=1 to l_kon do { UWAGA! Beta odniesi si do Cia^a 1 }
    begin
      { w jego uk^adzie tau, eta opisana jest vw obu cia^}
      d:=dl_wek(PO[K[i].c2,K[i].p2],PO[K[i].c1,K[i].p1]);
      bet:=k_wek(PO[K[i].c2,K[i].p2],PO[K[i].c1,K[i].p1]);
      alf:=K[i].ak+AL[K[i].c1];
      gam:=nor_k(pi+bet-alf);
      x:=d;
      st_k:=0; {0-poza zakresem; 1- kon. z odc.; 2- kon. z pkt.}

      if(K[i].z>0) then
        if((0<gam) and (gam<K[i].z)) then
          if(abs(gam)<pip) then
            begin
              st_k:=1;
            end;
    end;

```

```

        bet:=nor_k(alf - pip);
    end
    else st_k:=2;
if(K[i].z<0) then
    if((K[i].z<gam) and (gam<0)) then
        if(abs(gam)<pip) then
            begin
                st_k:=1;
                bet:=nor_k(alf + pip);
            end
        else st_k:=2;
    if(K[i].z=0) then
        st_k:=2;

    if(st_k=1)then
        begin
            x:=abs(d*sin(abs(gam)));
            if((d>K[i].dk) and not(pauza)) then
                st_k:=0;
        end;

    if((st_k=0)or((x-K[i].rk>=0)and(st_k>0))) then {brak kontaktu}
        begin
            KO[i].N:=0;
            KO[i].x:=-abs(x-K[i].rk);
            KO[i].T :=0;
            KO[i].bT[1]:=0;
            KO[i].bT[2]:=0;
        end;
    if((x-K[i].rk<0)and(st_k>0)) then {kontakt}
        begin
            KO[i].x:=abs(x-K[i].rk);
            KO[i].N:=R_mat(K[i].mat,KO[i].x,KO[i].vw.n);
            KO[i].b[1]:=bet;
            KO[i].b[2]:=nor_k(bet+pi);
        end;

        KO[i].vw.w.x:=VE[K[i].c1,K[i].p11].x-VE[K[i].c2,K[i].p_r].x;
        KO[i].vw.w.y:=VE[K[i].c1,K[i].p11].y-VE[K[i].c2,K[i].p_r].y;
        KO[i].vw.m:=dl_wek(p00,KO[i].vw.w);
        KO[i].vw.a:=k_wek (p00,KO[i].vw.w);
        KO[i].vw.t:=cos(KO[i].vw.a-KO[i].b[1]+pip)*KO[i].vw.m;
        KO[i].vw.n:=cos(KO[i].vw.a-KO[i].b[1]+pi )*KO[i].vw.m;

for j:=1 to 2 do
begin
    KO[i].w[j].x:=KO[i].N*cos(KO[i].b[j]);
    KO[i].w[j].y:=KO[i].N*sin(KO[i].b[j]);
end;
end;

for i:=1 to l_cial do
begin
    ST[i].MO:=ST[i].Q*ST[i].S[1].w.x;
    ST[i].KO:=p00;
    for j:=1 to ST[i].l_k do
        begin
            nK:=ST[i].K[j].nK;
            zw:=ST[i].K[j].zw;
            ST[i].MO:=ST[i].MO+il_wek(ST[i].K[j].w,KO[nK].w[zw]);
            ST[i].KO.x:=ST[i].KO.x+KO[nK].w[zw].x;

```

```

        ST[i].KO.y:=ST[i].KO.y+KO[nK].w[zw].y;
    end;
    for j:=1 to l_cial do
        ST[i].MO:=ST[i].MO+ST[i].M[j]*MSP[j];
    end;
        ST[6].MO:=ST[6].MO - 0.21*Wymusz;
end;
{-----}
procedure Sil_tar;
var v,v1,v2,alf,bet,w,T,T_s,T_d,T_gr,F,dF : Double;
m : array[1..l_kon_c] of Double;
i,j,nK,tar,zw : integer;
begin
    for i:=1 to l_cial do
        begin
            ST[i].sig:=0;
            for j:=1 to ST[i].l_k do
                begin
                    {rozk^ad si^y czynnej}
                    nK:=ST[i].K[j].nK;
                    zw:=ST[i].K[j].zw;
                    alf:=ST[i].K[j].a+AL[i];
                    if(K[nK].wym=zw) then
                        begin
                            ST[i].K[j].m:=abs(K[nK].tar*KO[nK].N*Mat[K[nK].mat].m_d
                                *cos(KO[nK].b[zw]-alf));
                            ST[i].sig:=ST[i].sig+ST[i].K[j].m;
                        end
                    else
                        ST[i].K[j].m:=0;
                end;
            if (ST[i].sig>1e-6) then
                for j:=1 to ST[i].l_k do
                    begin
                        nK:=ST[i].K[j].nK;
                        zw:=ST[i].K[j].zw;
                        alf:=ST[i].K[j].a;
                        bet:=KO[nK].b[zw];
                        KO[nK].Ft[zw]:=ST[i].K[j].m/ST[i].sig
                            * ST[i].MO/ST[i].K[j].r* cos(alf-bet+pi);
                        if((nK=n_kon) and (zw=ST[n_cial].K[i_kon].zw)) then
                            et4:=ST[i].K[j].m/ST[i].sig;
                    end
                else
                    for j:=1 to ST[i].l_k do
                        begin
                            nK:=ST[i].K[j].nK;
                            zw:=ST[i].K[j].zw;
                            KO[nK].Ft[zw]:=0;
                            if((nK=n_kon) and (zw=ST[n_cial].K[i_kon].zw)) then
                                et4:=0;
                        end;
                    end;
                for i:=1 to l_kon do
                    KO[i].Ft[3]:=KO[i].Ft[K[i].wym];
{.....}
                for i:=1 to l_kon do
                    if(KO[i].x>0) then
                        begin
                            bet:=KO[i].b[1];
                            if(abs(KO[i].vw.t)<Mat[K[i].mat].v0_T) then

```

```

w:=abs (KO[i].vw.t/Mat[K[i].mat].v0_T)
else w:=1;

T_d :=Mat[K[i].mat].m_d*KO[i].N;
T_gr:=Mat[K[i].mat].m_s*KO[i].N;
dF:=Mat[K[i].mat].dF;
F:=KO[i].Ft[3];
if(abs(F)<T_gr) then
  T_s:=abs(F);
if((T_gr<=abs(F)) and (abs(F)<T_gr+dF)) then
  T_s:=(abs(F)-T_gr)*(T_d-T_gr)/dF+T_gr;
if(abs(F)>=T_gr+dF) then
  T_s:=T_d;

T:=-w*T_d*sign(KO[i].vw.t) - (1-w)*T_s*sign(F);
if(T<=0) then
  KO[i].bT[1]:=nor_k(bet+pip);
if(T>0) then
  KO[i].bT[1]:=nor_k(bet-pip);
KO[i].T:=abs(T);
KO[i].bT[2]:=nor_k(KO[i].bT[1]+pi);

if(i=n_kon) then
begin
  et1:=T;
  et2:=T_s;
  et3:=T_d;
  et5:=w;
end;
end
else
begin
  KO[i].T:=0;
  KO[i].bT[1]:=0;
  KO[i].bT[2]:=0;
  if(i=n_kon) then
  begin
    et1:=0;
    et2:=0;
    et3:=0;
    et5:=0;
  end;
end;

for i:=1 to l_cial do
  for j:=1 to ST[i].l_k do
    begin
      nK:=ST[i].K[j].nK;
      zw:=ST[i].K[j].zw;
      KO[nK].wT[zw].x:=KO[nK].T*cos(KO[nK].bT[zw]);
      KO[nK].wT[zw].y:=KO[nK].T*sin(KO[nK].bT[zw]);
    end;

for i:=1 to l_cial do
  for j:=1 to ST[i].l_k do
    begin
      ST[i].MO:=ST[i].MO+iil_wek(ST[i].K[j].w,
        KO[ST[i].K[j].nK].wT[ST[i].K[j].zw]);
      ST[i].KO.x:=ST[i].KO.x+KO[ST[i].K[j].nK].wT[ST[i].K[j].zw].x;
      ST[i].KO.y:=ST[i].KO.y+KO[ST[i].K[j].nK].wT[ST[i].K[j].zw].y;
    end;

```

```

end;
{-----}
procedure Row_ruchu (x : Double;
                     n : Integer;
                     y : vector;
                     var f : vector); far;
var i,j,oj,ii,jj,w,k,met : integer;
r12,r13,r1,r2,r3 : pkt;
begin
  for i:=1 to l_cial do begin AL[i]:=y[i]; OM[i]:=y[i+l_cial]; end;

  Obl_pkt(1);
  Mom_spr;
  Sil_kon;
  Sil_tar;
  if((KO[17].x>0)and(wymusz>0)) then
    ST[6].MO:=ST[6].MO-OM[6]*10;

  AA[1,1]:=ST[1].I+ST[2].Mas*il_sk(ST[2].S[2].w,ST[2].S[2].w)
            +ST[3].Mas*il_sk(ST[3].S[2].w,ST[3].S[2].w);
  AA[1,2]:=      ST[2].Mas*il_sk(ST[2].S[2].w,ST[2].S[1].w);
  AA[2,1]:=AA[1,2];
  AA[2,2]:=ST[2].I;

  AA[1,3]:=      ST[3].Mas*il_sk(ST[3].S[2].w,ST[3].S[1].w);
  AA[3,1]:=AA[1,3];
  AA[3,3]:=ST[3].I;

  BB[1]:=+ST[1].MO
         +ST[2].Q*ST[2].S[2].w.x+il_wek(ST[2].S[2].w,ST[2].KO)
         +ST[3].Q*ST[3].S[2].w.x+il_wek(ST[3].S[2].w,ST[3].KO)
         +sqr(OM[2])*ST[2].Mas*il_wek(ST[2].S[1].w,ST[2].S[2].w)
         +sqr(OM[3])*ST[3].Mas*il_wek(ST[3].S[1].w,ST[3].S[2].w);

  BB[2]:=+ST[2].MO
         +sqr(OM[1])*ST[2].Mas*il_wek(ST[2].S[1].w,ST[2].S[2].w);

  BB[3]:=+ST[3].MO
         +sqr(OM[1])*ST[3].Mas*il_wek(ST[3].S[1].w,ST[3].S[2].w);

  case met_uk_row of
    0 : for i:=1 to n_mac do EP[i]:=BB[i]/AA[i,i];
    1 : Gauss(3,AA,BB,EP,st_gau);
    2 : begin
          GaussJordan (3,Wyp_mac,EP_,st_gau);
          for i:=1 to 3 do EP[i]:=EP_[i];
        end;
    3 : GaussSeidel (3,AA,BB,max_it_g,eps_g,EP,it_gau,st_gau);
  end;

  for i:=n_mac+1 to l_cial do
    EP[i]:=ST[i].MO/ST[i].I;

  for i:=1 to l_cial do
    begin
      f[i]:=      OM[i];
      f[i+l_cial]:=EP[i];
    end;
  end;
{-----}
{-----}

```

```
{-----}
procedure Zerowanie;
var i,j : integer;
pPO : pkt;
begin
  for i:=1 to 2*l_cial do  Y[i]:=0;
  for i:=0 to l_cial do
    for j:=1 to l_pkt do
      begin
        PO[i,j]:=p00; VE[i,j]:=p00;
      end;
  for i:=1 to l_cial do
    begin
      AL[i]:=Y[i];
      OM[i]:=Y[i+l_cial];
      EP[i]:=0;
      MSp[i]:=0;
    end;
    AL[0]:=0;
  for i:=1 to l_kon do
    KO[i].N:=0;
  for i:=1 to l_cial do
    begin
      for j:=1 to l_stop do
        ST[i].S[j].c:=0;
      for j:=1 to ST[i].l_k do
        ST[i].K[j].nK:=0;
      for j:=1 to l_cial do
        begin
          ST[i].M[j]:=0;
          AA[i,j]:=0;
        end;
        BB[i]:=0;
      end;
      et1:=0; et2:=0; et3:=0; et4:=0;
      {ustawienia wykres^w}
    for i:=1 to 7 do
      begin
        Wyk[i].y0:=Win[i].y2-Win[i].y1-10; Wyk[i].y:=0;
        Wyk[i].xx:=0; Wyk[i].yy:=Wyk[i].y0;
      end;

      for i:=1 to l_out do
        OUT[i].dat:=0;
    end;

procedure Przepisz_zlicz;
var i,j : integer;
pPO : pkt;
begin
  for i:=0 to l_cial do
    begin
      ST[i]:=ST0[i];
      SP[i]:=SP0[i];
    end;

    Assign(Fgeo,file_geo);
    reset(Fgeo);
    if((IOResult=0)and(filesize(Fgeo)=(l_cial+1)*l_pkt)) then
      begin
        for i:=0 to l_cial do
```

```

        for j:=1 to l_pkt do
        begin
            Seek(Fgeo,i*l_pkt+j-1);
            read(Fgeo,P[i,j]);
        end;
        close(Fgeo);
    end
else
begin writeln('Plik '+file_geo+' niezadadowany');
    readln; end;

Assign(Fkon,file_kon);
reset(Fkon);
if((IOResult=0)and(filesize(Fkon)=l_kon)) then
begin
    for i:=1 to l_kon do
    begin
        Seek(Fkon,i-1);
        read(Fkon,K[i]);
    end;
    close(Fkon);
end
else
begin writeln('Plik '+file_kon+' niezadadowany');
    readln; end;

Assign(Fmat,file_mat);
reset(Fmat);
if((IOResult=0)and(filesize(Fmat)=l_mat)) then
begin
    for i:=1 to l_mat do
    begin
        Seek(Fmat,i-1);
        read(Fmat,MAT[i]);
    end;
    close(Fmat);
end
else
begin writeln('Plik '+file_mat+' niezadadowany');
    readln; end;

for i:=0 to l_cial do {zliczanie pkt-w}
begin
    ST[i].l_p:=l_pkt;
    for j:=l_pkt downto 1 do
        if(P[i,j].k<0) then
            ST[i].l_p:=j-1;
end;

end;

procedure WYP_PKT;
var i,j : integer;
pPO,MAP : pkt;
begin
    for i:=1 to l_kon do
    begin
        P[K[i].c1,K[i].p11].k:=1; {aktywacja pkt-w kontakt-w}
        P[K[i].c1,K[i].p12].k:=1; {dla nich liczona pr@dko +}
        P[K[i].c2,K[i].p2].k:=1; { i poolenia}
        P[K[i].c2,K[i].p_r].k:=1;
    end;

```

```

    end;
for i:=1 to l_cial do
begin
  P[i,2].k:=1;
  P[P[i,1].pc,P[i,1].pp].k:=1;
end;

Assign(Fmap,file_map);
reset(Fmap);
for i:=0 to l_cial do {wype nia tab. punkt^w}
  for j:=1 to ST[i].l_p do
    begin
      if (P[i,j].pp=0) then
        begin
          Seek(Fmap,P[i,j].LM-1);
          read(Fmap,MAP);
          P[i,j].a:=k_wek(p00,MAP);
          P[i,j].r:=dl_wek(p00,MAP);
          PO[i,j].x:=p00.x+P[i,j].r*cos(P[i,j].a);
          PO[i,j].y:=p00.y+P[i,j].r*sin(P[i,j].a);
        end
      else
        begin
          Seek(Fmap,P[i,j].LM-1);
          read(Fmap,MAP);
          pPO:=PO[P[i,j].pc,P[i,j].pp];
          P[i,j].a:=k_wek(pPO,MAP);
          P[i,j].r:=dl_wek(pPO,MAP);
          PO[i,j].x:=pPO.x+P[i,j].r*cos(P[i,j].a+AL[i]);
          PO[i,j].y:=pPO.y+P[i,j].r*sin(P[i,j].a+AL[i]);
        end;
    end;
close(Fmap);

for i:=0 to l_cial do {uwzgl@dnia poprawki}
  for j:=1 to l_pkt do
    begin
      if (P[i,j].pp=0) then
        begin
          P[i,j].a:=P[i,j].a+P[i,j].da;
          P[i,j].r:=P[i,j].r+P[i,j].dr;
          P[i,j].ab:=P[i,j].a;
          P[i,j].rb:=P[i,j].r;
          PO[i,j].x:=p00.x+P[i,j].r*cos(P[i,j].a)+P[i,j].dx;
          PO[i,j].y:=p00.y+P[i,j].r*sin(P[i,j].a)+P[i,j].dy;
        end
      else
        begin
          pPO:=PO[P[i,j].pc,P[i,j].pp];
          P[i,j].da:=P[i,j].da+P[P[i,j].pc,P[i,j].pp].da;
          P[i,j].a:=P[i,j].a+P[i,j].da;
          P[i,j].r:=P[i,j].r+P[i,j].dr;
          PO[i,j].x:=pPO.x+P[i,j].r*cos(P[i,j].a+AL[i])+P[i,j].dx;
          PO[i,j].y:=pPO.y+P[i,j].r*sin(P[i,j].a+AL[i])+P[i,j].dy;
          P[i,j].ab:=k_wek(PO[i,1],PO[i,j]);
          P[i,j].rb:=dl_wek(PO[i,1],PO[i,j]);
        end;
    end;
end;

procedure WYP_STRUK;

```

```

var i,j,pk,pc,pp : integer;
begin
  for i:=1 to l_cial do {wype^nia tab.hierarchii }
  begin
    ST[i].l_st:=1;           { . stopnie hierarchii}
    ST[i].S[1].c:=i;
    ST[i].S[1].p:=2;
    ST[i].S[1].r:=dl_wek(PO[i,1],PO[i,2]);
    ST[i].S[1].a:=k_wek (PO[i,1],PO[i,2]);
    ST[i].M[i]:=1;
    pc:=P[i,1].pc;
    pp:=P[i,1].pp;
    for j:=2 to l_stop do
      if pc>0 then
        begin
          ST[i].l_st:=ST[i].l_st+1;
          ST[i].S[j].c:=pc;
          ST[i].S[j].p:=pp;
          ST[i].S[j].r:=dl_wek(PO[pc,1],PO[pc,pp]);
          ST[i].S[j].a:=k_wek (PO[pc,1],PO[pc,pp]);
          ST[pc].M[i]:=-1;
          pc:=P[pc,1].pc;
          pp:=P[pc,1].pp;
        end;
    ST[i].l_k:=0;
    for j:=1 to l_kon do { . kontakty}
      if ((K[j].c1=i)or(K[j].c2=i)) then
        begin
          ST[i].l_k:=ST[i].l_k+1;
          ST[i].K[ST[i].l_k].nK:=j;
          if(K[j].c1=i)then
            begin ST[i].K[ST[i].l_k].zw:=1;
            pK:=K[j].p11; end
          else
            begin ST[i].K[ST[i].l_k].zw:=2;
            pK:=K[j].p_r; end;
          ST[i].K[ST[i].l_k].a:=k_wek (PO[i,1],PO[i,pK]);
          ST[i].K[ST[i].l_k].r:=dl_wek(PO[i,1],PO[i,pK]);
          ST[i].K[ST[i].l_k].p:=pK;
        end;
    ST[i].Q:=ST[i].Mas*g;           {obliczanie Q }
  end;
end;

procedure Inicjuj(inic : integer);
var i,j,pk,pc,pp : integer;
r : double;
f : vector;
pPO : pkt;
begin
  if (inic=0) then
  begin
    czysc_:=false;
    n_cial:=4;
    i_kon:=5; n_kon:=12;
    f1:='k'; ff1:='k';
    f2:='t'; ff2:='p';
    n_pop:=3;
    pauza:=false;
  end;

```

```
start:=true;
Ekr.x:=100; Ekr.y:=145; Ekr.s:=700;

Assign(Fpop,file_pop);
reset(Fpop);
if((IOResult=0)and(filesize(Fpop)=l_pop)) then
begin
  for i:=1 to l_pop do
  begin
    Seek(Fpop,i-1);
    read(Fpop,POP[i]);
  end;
  close(Fpop);
end
else
begin
  writeln('Plik '+file_pop+' nieza^adowany');
  readln;
end;

Assign(Fmap,file_map);
reset(Fmap);
if not((IOResult=0)and(filesize(Fmap)=l_LM)) then
begin
  writeln('Plik '+file_map+' nieza^adowany');
  readln;
end;

Assign(F_in,File_input);
reset(F_in);
if not(IOResult=0) then
begin
  writeln('Plik '+file_input+' nieza^adowany');
  readln;
end;
close(F_in);

GraphDriver := VGA;
GraphMode := VGAMED;
InitGraph(GraphDriver,GraphMode,'c:\tp\bgi');
{InitEvents;}
end;
{-----za kazdym razem-----}

Zerowanie;
Przepisz_zlicz;
Wpisz_dane;

Assign(F_in,File_input);
reset(F_in);
Assign(Fout,File_output);
rewrite(Fout);
for i:=1 to l_out do write(Fout,OUT[i].tx);
writeln(Fout);

t_in:=0;
t0:=0;
t1:=t0;
t_wyk:=0;
h:=dt;
GetTime(t[1].h,t[1].m,t[1].s,t[1].ms);
```

```
fl:=true;
Vpg:=1;

POPRAW;           {aktualizacja poprawek}
WYP_PKT;
WYP_STRUK;

for i:=1 to l_kon do
begin
  K[i].ak:= k_wek(PO[K[i].c1,K[i].p11],PO[K[i].c1,K[i].p12]);
  K[i].dk:=dl_wek(PO[K[i].c1,K[i].p11],PO[K[i].c1,K[i].p12]);
  K[i].rk:=dl_wek(PO[K[i].c2,K[i].p2],PO[K[i].c2,K[i].p_r]);
end;

Sil_kon;

for i:=1 to l_cial do
begin
  SP[i].m0:=SP[i].m0+SP[i].dm0;
  SP[i].m0:=SP[i].m0+SP[i].dm0;
end;

end;

procedure klawiatura;
var i,j : integer;
begin
  {GetMouseEvent(Event);
  Mouse:=MouseWhere;
  wymusz:=(Mouse.x+Mouse.y)*sk_wymusz; }
  if (keypressed) then klaw:=readkey;
  case klaw of
    'p': begin pauza:=not(pauza); klaw:=' '; end;
    's': begin
      start:=true;
      Inicjuj(1);
      klaw:=' ';
    end;
    'S': begin
      Assign(Fsta,file_start);
      reset(Fsta);
      if (IOResult=0) then
        begin
          read(Fsta,tY_start);
          close(Fsta);
          t0:=tY_start.t;
          Y:=tY_start.Y;
          reset(F_in);
          sound(1000);
          delay(50);
          nosound;
        end;
      klaw:=' ';
    end;
    'Z': begin
      tY_start.t:=t0;
      tY_start.Y:=Y;
      Assign(Fsta,file_start);
      rewrite(Fsta);
      write(Fsta,tY_start);
      close(Fsta);
    end;
  end;
end;
```

```

        klaw:=' ';
        sound(1000);
        delay(50);
        nosound;
    end;
'0': begin
    czysc_:=not(czysc_);
    klaw:=' ';
end;
'4' : begin Ekr.x:=Ekr.x+100; klaw:=' '; end;
'6' : begin Ekr.x:=Ekr.x-100; klaw:=' '; end;
'8' : begin Ekr.y:=Ekr.y+100; klaw:=' '; end;
'2' : begin Ekr.y:=Ekr.y-100; klaw:=' '; end;
'+' : begin Ekr.s:=Ekr.s*2;   klaw:=' '; end;
'-' : begin Ekr.s:=Ekr.s/2;   klaw:=' '; end;
'e' : begin
    ff1:=f1;
    case f1 of
        ' ' : f1:='c';
        'c' : f1:='k';
        'k' : f1:=' ';
    end;
    klaw:=' ';
end;
'x','t','d',#59 : begin
    ff2:=f2;
    f2:=klaw;
    klaw:=' ';
end;

#81 :begin
    i_kon:=i_kon-1;
    if(i_kon=0) then
        i_kon:=ST[n_cial].l_k;
    if(i_kon>ST[n_cial].l_k) then
        i_kon:=1;
    n_kon:=ST[n_cial].K[i_kon].nK;
    klaw:=' ';
end;
#73 :begin
    i_kon:=i_kon+1;
    if(i_kon>ST[n_cial].l_k) then
        i_kon:=1;
    n_kon:=ST[n_cial].K[i_kon].nK;
    klaw:=' ';
end;

#79 :begin
    n_cial:=n_cial-1;
    if(n_cial=0) then
        n_cial:=1_cial;
    if(i_kon>ST[n_cial].l_k) then
        i_kon:=1;
    n_kon:=ST[n_cial].K[i_kon].nK;
    klaw:=' ';
end;
#71 :begin
    n_cial:=n_cial+1;
    if(n_cial>1_cial) then
        n_cial:=1;
    if(i_kon>ST[n_cial].l_k) then

```

```

        i_kon:=1;
        n_kon:=ST[n_cial].K[i_kon].nK;
        klaw:=' ';
    end;
end;
begin
procedure Opoznenie;
var i,j : integer;
begin
    GetTime(t[2].h,t[2].m,t[2].s,t[2].ms);
    for i:=1 to 2 do
        begin
            t_[i].h:=t[i].h; t_[i].m:=t[i].m;
            t_[i].s:=t[i].s; t_[i].ms:=t[i].ms;
        end;
    t_r:=((t_[2].h-t_[1].h)*3600+(t_[2].m-t_[1].m)*60
        +(t_[2].s- t_[1].s)+(t_[2].ms-t_[1].ms)/100);
    if (t_r<t1) then
        delay(round(t1-t_r)*100);
end;

procedure InOut;
var i,code: integer;
s1,s2           : string[8];
begin
    Wpisz_dane;
    while ((t0>=t_in) and not(Eof(F_in))) do
        begin
            read(F_in,s1);
            readln(F_in,s2);
            for i:=1 to 8 do
                if(s1[i]=' ') then
                    delete(s1,i,8);
            for i:=1 to 8 do
                if(s2[i]=' ') then
                    delete(s2,i,8);
            val(s1,t_in,code);
            val(s2,Wymusz,code);
            t_in:=t_in;
        end;

    if(t0<=t_max) then
        for i:=1 to l_out do
            write(Fout,OUT[i].dat:8:4);
            writeln(Fout);
end;

{-----}
begin
Inicjuj(0);
repeat
    InOut;
    Klawiatura;
    Obl_pkt(0);
    Rysuj(f1,f2);
    if not(pauza) then
        begin
            t1:=t0+dt;
            Fehlberg1(t0,t1,2*l_cial,Y,Row_ruchu,mh,eps,eta,max_it,st_cal,h);

```

```
    Opoznenie;  
  end;  
until (klaw='q');  
close(F_in);  
close(Fout);  
closeGraph;  
end.
```

Program Geom_4;

```
Uses Dane_4;
```

```
Const  
MAPO : array[1..1 LM] of pkt=  
((x:0.00691 ; y:0.0189 ),  
(x:0.0239 ; y:0.0179 ),  
(x:0.0173 ; y:0.0254 ),  
(x:0.0173 ; y:0.0214 ),  
(x:0.0295 ; y:0.0173 ),  
(x:0.0163 ; y:0.0635 ),  
(x:0.01 ; y:0.0848 ),  
(x:0.0188 ; y:0.087 ),  
(x:0.0254 ; y:0.0682 ),  
(x:0.0352 ; y:0.0537 ),  
{10}  
(x:0.0374 ; y:0.0553 ),  
(x:0.0389 ; y:0.0512 ),  
(x:0.0358 ; y:0.0732 ),  
(x:0.159 ; y:0.106 ),  
(x:0.157 ; y:0.112 ),  
(x:0.0349 ; y:0.0782 ),  
(x:0.0333 ; y:0.109 ),  
(x:0.0236 ; y:0.123 ),  
(x:0.0198 ; y:0.116 ),  
(x:0.0126 ; y:0.12 ),  
{20}  
(x:0.011 ; y:0.104 ),  
(x:0.0261 ; y:0.0764 ),  
(x:0.0895 ; y:0.101 ),  
(x:0.154 ; y:0.108 ),  
(x:0.138 ; y:0.104 ),  
(x:0.14 ; y:0.0952 ),  
(x:0.141 ; y:0.0899 ),  
(x:0.106 ; y:0.0449 ),  
(x:0.0568 ; y:0.0481 ),  
(x:0.0502 ; y:0.0449 ),  
{30}  
(x:0.106 ; y:0.0408 ),  
(x:0.112 ; y:0.0233 ),  
(x:0.121 ; y:0.0233 ),  
(x:0.124 ; y:0.039 ),  
(x:0.157 ; y:0.0364 ),  
(x:0.158 ; y:0.044 ),  
(x:0.0951 ; y:0.0512 ),  
(x:0.097 ; y:0.0745 ),  
(x:0.0904 ; y:0.0745 ),  
(x:0.0876 ; y:0.0512 ),  
{40}  
(x:0.0499 ; y:0.053 ),  
(x:0.0933 ; y:0.071 ),  
(x:0.103 ; y:0.0745 ),  
(x:0.0644 ; y:0.0647 ),  
(x:0.0663 ; y:0.0572 ),  
(x:0.153 ; y:0.0858 ),  
(x:0.165 ; y:0.0867 ),  
(x:0.165 ; y:0.0921 ),  
(x:0.154 ; y:0.0921 ),  
(x:0.156 ; y:0.0405 ),  
{50}  
(x:0.153 ; y:0.0336 ),  
(x:0.141 ; y:0.0867 ),  
(x:0.148 ; y:0.089 ),  
(x:0.158 ; y:0.0415 ),  
(x:0.179 ; y:0.0377 ),  
(x:0.179 ; y:0.0408 ),  
(x:0.179 ; y:0.044 ),  
(x:0.152 ; y:0.0613 ),  
(x:0.00628; y:0.00063),  
(x:0.267 ; y:0.00566),  
{60}
```

```

(x:0.267 ; y:-0.0063),
(x:0.507 ; y:-0.0063),
(x:0.507 ; y:0.015 ),
(x:0.267 ; y:0.0189 ),
(x:0.116 ; y:0.0141 ),
(x:0.116 ; y:0.0192 ),
(x:0.116 ; y:0.0226 ),
(x:-0.058 ; y:0.0408 ),
(x:-0.018 ; y:0.0547 ),
(x:-0.0082; y:0.0361 ),
{70}
(x:-0.0079; y:0.196 ),
(x:-0.0079; y:0.181 ),
(x:0.0267 ; y:0.178 ),
(x:-0.042 ; y:0.178 ),
(x:-0.0031; y:0.0408 ),
(x:-0.063 ; y:0.0456 ),
(x:-0.063 ; y:0.0361 ),
(x:-0.0031; y:0.0314 ),
(x:-0.063 ; y:0.171 ),
(x:0.0801 ; y:0.171 ),
{80}
(x:0.173 ; y:0.0547 ),
(x:0.173 ; y:0.0447 ),
(x:0.188 ; y:0.0447 ),
(x:0.1587 ; y:0.1 ),
(x:0.1587 ; y:0.098 ),
(x:0.174 ; y:0.098 ),
(x:0.127 ; y:0.047 ),
(x:0.127 ; y:0.055 ),
(x:0.133 ; y:0.060 ),
(x:0.130 ; y:0.069 ),
{90}
(x:0.126 ; y:0.068 ),
(x:0.121 ; y:0.067 ),
(x:0.120 ; y:0.048 ),
(x:0.119 ; y:0.084 ),
(x:0.122 ; y:0.085 ),
(x:0.125 ; y:0.086 ),
(x:0.122 ; y:0.084 ),
(x:0.123 ; y:0.080 ),
(x:0.124 ; y:0.075 ),
(x:0.165 ; y:0.089 ),
{100}
(x:0.165 ; y:0.092 ),
(x:0.162 ; y:0.089 ),
(x:0.147 ; y:0.066 ),
(x:0.137 ; y:0.065 ),
(x:0.133 ; y:0.064 ),
(x:0.01 ; y:0.00063),
(x:0.01 ; y:-0.002 ),
(x:0.49 ; y:-0.0163),
(x:0.49 ; y:-0.022 ),
(x:-0.013 ; y:0.0508 ),
{110}
(x:-0.013 ; y:0.048 ),
(x:0.017 ; y:0.123 ),
(x:0.146 ; y:0.089));

```

```

P0 : array[0..l_cial,1..l_pkt] of geom={k:-1 koniec ciala}
{----- inertial space}
((k:0; LM:61; pc:0; pp:0; r_:0), { o~obrotu cia^a}
 (k:0; LM:29; pc:0; pp:0; r_:0), { źodek masy (dla ciala 0 bez znaczenia)}
 (k:0; LM:24; pc:0; pp:0; r_:0),
 (k:0; LM:68; pc:0; pp:0; r_:0),
 (k:0; LM:79; pc:0; pp:0; r_:0),

```

```

(k:0; LM:80; pc:0; pp:5; r_:1),
(k:0; LM:81; pc:0; pp:0; r_:0),
(k:0; LM:82; pc:0; pp:7; r_:1),
(k:0; LM:83; pc:0; pp:8; r_:1),
(k:0; LM:84; pc:0; pp:0; r_:0),
{10}
(k:0; LM:85; pc:0; pp:10;r_:1),
(k:0; LM:86; pc:0; pp:11;r_:1),
(k:0; LM:107;pc:0; pp:0;r_:0),
(k:0; LM:106;pc:0; pp:13;r_:0),

```

```

(k:0; LM:109;pc:0; pp:0;r_:0),
(k:0; LM:108;pc:0; pp:15;r_:0),
(k:0; LM:110;pc:0; pp:1;r_:0),
(k:0; LM:111;pc:0; pp:17;r_:0),
(k:-1),(),(),(),(),(),(),(),(),(),
{----- figura}
((k:0; LM:29; pc:0; pp:2; r_:0),
(k:0; LM:28; pc:1; pp:1; r_:0),
(k:0; LM:30; pc:1; pp:2; r_:0),
(k:0; LM:31; pc:1; pp:3; r_:1),
(k:0; LM:32; pc:1; pp:4; r_:1),
(k:0; LM:33; pc:1; pp:5; r_:1),
(k:0; LM:34; pc:1; pp:6; r_:1),
(k:0; LM:35; pc:1; pp:7; r_:1),
(k:0; LM:36; pc:1; pp:8; r_:1),
(k:0; LM:87; pc:1; pp:9; r_:1),
{10}
(k:0; LM:88; pc:1; pp:10; r_:1),
(k:0; LM:89; pc:1; pp:11; r_:1),
(k:0; LM:90; pc:1; pp:12; r_:1),
(k:0; LM:91; pc:1; pp:13; r_:1),
(k:0; LM:92; pc:1; pp:14; r_:1),
(k:0; LM:93; pc:1; pp:15; r_:1),
(k:0; LM:37; pc:1; pp:16; r_:1),
(k:0; LM:38; pc:1; pp:17; r_:1),
(k:0; LM:39; pc:1; pp:18; r_:1),
(k:0; LM:40; pc:1; pp:19; r_:1),
{20}
(k:0; LM:41; pc:1; pp:20; r_:1),
(k:0; LM:30; pc:1; pp:21; r_:1),
(k:0; LM:50; pc:1; pp:2; r_:0),
(k:0; LM:42; pc:1; pp:2; r_:0),
(k:0; LM:95; pc:1; pp:14; r_:1),
(k:0; LM:94; pc:1; pp:25; r_:0),
(k:0; LM:96; pc:1; pp:26; r_:1),
(k:-1),(),(),
{----- dwignia rep.}
((k:0; LM:42; pc:1; pp:24; r_:0),
(k:0; LM:43; pc:2; pp:1; r_:0),
(k:0; LM:44; pc:2; pp:2; r_:0),
(k:0; LM:45; pc:2; pp:3; r_:1),
(k:0; LM:46; pc:2; pp:4; r_:1),
(k:0; LM:47; pc:2; pp:5; r_:1),
(k:0; LM:48; pc:2; pp:6; r_:1),
(k:0; LM:49; pc:2; pp:7; r_:1),
(k:0; LM:44; pc:2; pp:8; r_:1),
(k:0; LM:100;pc:2; pp:2; r_:0),
(k:0; LM:102;pc:2; pp:10; r_:0),
(k:0; LM:98; pc:2; pp:2; r_:0),
(k:0; LM:99; pc:2; pp:12; r_:0),
(k:0; LM:95; pc:2; pp:12; r_:0),
(k:0; LM:101;pc:2; pp:10; r_:0),
(k:0; LM:113;pc:2; pp:2; r_:0),
(k:-1),(),(),(),(),(),(),(),(),(),
{----- bijnik}
((k:0; LM:50; pc:1; pp:23; r_:0),
(k:0; LM:58; pc:3; pp:1; r_:0),
(k:0; LM:51; pc:3; pp:2; r_:0),
(k:0; LM:52; pc:3; pp:3; r_:1),
(k:0; LM:53; pc:3; pp:4; r_:1),
(k:0; LM:54; pc:3; pp:5; r_:1),
(k:0; LM:56; pc:3; pp:6; r_:1),
(k:0; LM:55; pc:3; pp:7; r_:1),
(k:0; LM:51; pc:3; pp:8; r_:1),
(k:0; LM:57; pc:3; pp:7; r_:0),
(k:0; LM:103;pc:3; pp:2; r_:0),
(k:0; LM:104;pc:3; pp:11; r_:1),
(k:0; LM:105;pc:3; pp:12; r_:0),

```

```

(k:-1), (), (), (), (), (), (), (), (), (), (), (), (), (), (), (), (), (),
{----- m^otek}
((k:0; LM:24; pc:0; pp:3; r_:0),
(k:0; LM:23; pc:4; pp:1; r_:0),
(k:0; LM:10; pc:4; pp:2; r_:0),
(k:0; LM:11; pc:4; pp:2; r_:0),
(k:0; LM:12; pc:4; pp:3; r_:1),
(k:0; LM:13; pc:4; pp:5; r_:1),
(k:0; LM:14; pc:4; pp:6; r_:1),
(k:0; LM:15; pc:4; pp:7; r_:1),
(k:0; LM:16; pc:4; pp:8; r_:1),
(k:0; LM:17; pc:4; pp:6; r_:1),
{10}
(k:0; LM:18; pc:4; pp:10; r_:1),
(k:0; LM:20; pc:4; pp:11; r_:1),
(k:0; LM:19; pc:4; pp:2; r_:0),
(k:0; LM:21; pc:4; pp:12; r_:1),
(k:0; LM:22; pc:4; pp:14; r_:1),
(k:0; LM:10; pc:4; pp:15; r_:1),
(k:0; LM:25; pc:4; pp:2; r_:0),
(k:0; LM:26; pc:4; pp:17; r_:1),
(k:0; LM:27; pc:4; pp:18; r_:0),
(k:0; LM:112; pc:4; pp:13; r_:0),
(k:-1), (), (), (), (), (), (), (), (), (), (),
{----- t^umik}
((k:0; LM:68; pc:0; pp:4; r_:0),
(k:0; LM:69; pc:5; pp:1; r_:0),
(k:0; LM:70; pc:5; pp:2; r_:0),
(k:0; LM:76; pc:5; pp:3; r_:0),
(k:0; LM:77; pc:5; pp:4; r_:1),
(k:0; LM:78; pc:5; pp:5; r_:1),
(k:0; LM:75; pc:5; pp:6; r_:1),
(k:0; LM:71; pc:5; pp:3; r_:1),
(k:0; LM:72; pc:5; pp:8; r_:0),
(k:0; LM:74; pc:5; pp:8; r_:1),
(k:0; LM:73; pc:5; pp:10; r_:1),
(k:0; LM:71; pc:5; pp:11; r_:1),
(k:0; LM:76; pc:5; pp:7; r_:1),
(k:0; LM:4; pc:5; pp:6; r_:1),
(k:0; LM:3; pc:5; pp:14; r_:1),
(k:-1), (), (), (), (), (), (), (), (), (), (), (),
{----- klawisz}
((k:0; LM:61; pc:0; pp:1; r_:0),
(k:0; LM:60; pc:6; pp:1; r_:0),
(k:0; LM:62; pc:6; pp:2; r_:0),
(k:0; LM:63; pc:6; pp:3; r_:1),
(k:0; LM:65; pc:6; pp:4; r_:1),
(k:0; LM:5; pc:6; pp:5; r_:1),
(k:0; LM:1; pc:6; pp:6; r_:1),
(k:0; LM:59; pc:6; pp:7; r_:1),
(k:0; LM:2; pc:6; pp:7; r_:1),
(k:0; LM:7; pc:6; pp:6; r_:1),
{10}
(k:0; LM:8; pc:6; pp:10; r_:1),
(k:0; LM:9; pc:6; pp:11; r_:1),
(k:0; LM:6; pc:6; pp:12; r_:1),
(k:0; LM:66; pc:6; pp:5; r_:1),
(k:0; LM:67; pc:6; pp:14; r_:0),
(k:0; LM:61; pc:6; pp:8; r_:1),
(k:0; LM:62; pc:6; pp:16; r_:1),
(k:-1), (), (), (), (), (), (), (), (), (), ());
K0      : array[1..1_kon]          of kontakty_cons=
((c1:1; p11:5; p12:6; z:-2;     c2:6; p2:14; p_r:15; rk:0;
tx:'pilot-stopka'; tar:1; wym:1; mat:2),
(c1:0; p11:8; p12:9; z:-2;     c2:3; p2:7;  p_r:10; rk:0; tx:'bij.-pupka
'; tar:0; wym:1; mat:1),

```

```

(c1:1; p11:13;p12:12;z:2;      c2:3; p2:12; p_r:13; rk:0; tx:'bij.-figura
'; tar:0; wym:1; mat:1),
(c1:3; p11:5; p12:6; z:2;      c2:2; p2:10; p_r:11; rk:0;
tx:'bij.-d<<w.rep.'; tar:0; wym:1; mat:1),
(c1:3; p11:4; p12:3; z:-1.793;c2:4; p2:18; p_r:19; rk:0; tx:'bar.-bijnik
1'; tar:1; wym:2; mat:3),
(c1:3; p11:4; p12:5; z:2.824; c2:4; p2:18; p_r:19; rk:0; tx:'bar.-bijnik
2'; tar:1; wym:2; mat:3),
(c1:2; p11:16;p12:9; z:-2;      c2:4; p2:18; p_r:19; rk:0;
tx:'bar.-d<<w.rep.'; tar:1; wym:2; mat:3),
(c1:0; p11:11;p12:12;z:-2;      c2:2; p2:10; p_r:15; rk:0;
tx:'d<<w.rep.-ogr.'; tar:0; wym:1; mat:1),
(c1:1; p11:15;p12:13;z:2;      c2:2; p2:12; p_r:13; rk:0;
tx:'d<<w.-fig.dol.'; tar:0; wym:1; mat:1),
(c1:1; p11:26;p12:27;z:-2;      c2:2; p2:12; p_r:14; rk:0;
tx:'d<<w.-fig.g^r.'; tar:0; wym:1; mat:1),
(c1:0; p11:6; p12:5; z:2;      c2:4; p2:13; p_r:20; rk:0;
tx:'m^ot.-struna '; tar:0; wym:1; mat:4),
(c1:6; p11:11;p12:12;z:2;      c2:4; p2:4; p_r:3; rk:0;
tx:'m^ot.-chwytek'; tar:1; wym:2; mat:5),
(c1:0; p11:6; p12:5; z:-2;      c2:5; p2:8; p_r:9; rk:0;
tx:'t^um.-struna '; tar:0; wym:1; mat:1),
(c1:5; p11:7; p12:4; z:-2;      c2:0; p2:17; p_r:18; rk:0; tx:'tlum.-ogr.
'; tar:0; wym:1; mat:1),
(c1:6; p11:7; p12:9; z:2;      c2:5; p2:15; p_r:14; rk:0; tx:'ylka-klaw.
'; tar:0; wym:1; mat:1),
(c1:6; p11:8; p12:1; z:-2;      c2:0; p2:13; p_r:14; rk:0; tx:'oparcie
klaw.>'; tar:0; wym:1; mat:1),
(c1:6; p11:3; p12:1; z:2;      c2:0; p2:15; p_r:16; rk:0; tx:'dno
klawisz.>'; tar:0; wym:1; mat:1));
dane_starowe : start_data=
(t:0; y:(0,0,0,0,0,0, 0,0,0,0,0,0));
var
i,j,n : integer;
p_:pkt;
dec : char;
begin
Assign(Fmap,File_map);
rewrite(Fmap);
writeln('Mapa punkt^w');
for i:=1 to l_LM do
begin
  write(Fmap,MAP0[i]);
end;
writeln('Liczba rekord^w : ',filesize(Fmap));
close(Fmap);

Assign(Fgeo,File_Geo);
rewrite(Fgeo);
writeln('Geometria uk adu');
for i:=0 to l_cial do
  for j:=1 to l_pkt do
    begin
      write(Fgeo,P0[i,j]);
    end;

```

```
writeln('Liczba rekordów : ', filesize(Fgeo));
close(Fgeo);

Assign(Fkon, File_kon);
rewrite(Fkon);
writeln('Kontakty');
for i:=1 to l_kon do
begin
  write(Fkon, K0[i]);
end;
writeln('Liczba rekordów : ', filesize(Fkon));
close(Fkon);

writeln('Czy wyzerowac plik startowy? (t/n)');
readln(dec);
if(dec='t') then
begin
  Assign(Fsta, File_start);
  writeln('Dane stare');
  rewrite(Fsta);
  write(Fsta, Dane_starowe);
  writeln('czas startu', Dane_starowe.t);
  close(Fsta);
  readln;
end;
end.
```

Program Matpop_4;

Uses Dane_4;

const

```
MAT0 : array[1..l_mat] of material=
((naz:'filc'; v0_N:-0.01; m_s:0.7; m_d:0.57; v0_T:0.01; dF:0.01;
T:((x: 0; yob: 0; aob: 0; yod: 0; aod: 0),
(x: 0.0002; yob: 2; aob: 10000; yod: 0; aod: 0),
(x: 0.0004; yob: 4; aob: 10000; yod: 0; aod: 0),
(x: 0.0006; yob: 7; aob: 15000; yod: 1; aod: 5000),
(x: 0.0008; yob: 11; aob: 20000; yod: 2; aod: 5000),
(x: 0.001 ; yob: 18; aob: 35000; yod: 3; aod: 5000),
(x: 0.0012; yob: 27; aob: 45000; yod: 5; aod: 10000),
(x: 0.0014; yob: 40; aob: 65000; yod: 7; aod: 10000),
(x: 0.0016; yob: 55; aob: 75000; yod: 9; aod: 10000),
(x: 0.0018; yob: 85; aob: 150000; yod: 11; aod: 10000),
(x: 0.002 ; yob: 140; aob: 275000; yod: 13; aod: 10000),
(x: 0.0022; yob: 225; aob: 425000; yod: 15; aod: 10000),
(x: 0.0024; yob: 370; aob: 725000; yod: 17; aod: 10000),
(x: 0.0026; yob: 370; aob: 0; yod: 17; aod: 0),
(x: 0.0028; yob: 370; aob: 0; yod: 17; aod: 0))),,
(naz:'stopka'; v0_N:-0.01; m_s:0.26; m_d:0.21; v0_T:0.01; dF:0.01;
T:((x: 0 ; yob: 0; aob: 0; yod: 0; aod: 0),
(x: 0.0001 ; yob: 3; aob: 30000; yod: 0; aod: 0),
(x: 0.0002 ; yob: 5; aob: 20000; yod: 0; aod: 0),
(x: 0.0003 ; yob: 8; aob: 30000; yod: 0; aod: 0),
(x: 0.0004 ; yob: 12; aob: 40000; yod: 1; aod: 10000),
(x: 0.0005 ; yob: 16; aob: 40000; yod: 2; aod: 10000),
(x: 0.0006 ; yob: 22; aob: 60000; yod: 3; aod: 10000),
(x: 0.0007 ; yob: 30; aob: 80000; yod: 4; aod: 10000),
(x: 0.0008 ; yob: 40; aob: 100000; yod: 5; aod: 10000),
(x: 0.0009 ; yob: 55; aob: 150000; yod: 6; aod: 10000),
(x: 0.001 ; yob: 73; aob: 180000; yod: 7; aod: 10000),
(x: 0.0011 ; yob: 94; aob: 210000; yod: 8; aod: 10000),
(x: 0.0012 ; yob: 116; aob: 220000; yod: 10; aod: 20000),
(x: 0.0013 ; yob: 145; aob: 290000; yod: 12; aod: 20000),
(x: 0.0014 ; yob: 145; aob: 0; yod: 12; aod: 0))),
(naz:'bary^ka'; v0_N:-0.01; m_s:0.7; m_d:0.57; v0_T:0.01; dF:0.01;
T:((x: 0 ; yob: 0; aob: 0; yod: 0; aod: 0),
(x: 0.0001 ; yob: 2; aob: 20000; yod: 0; aod: 0),
(x: 0.0002 ; yob: 6; aob: 40000; yod: 0; aod: 0),
(x: 0.0003 ; yob: 9; aob: 30000; yod: 0; aod: 0),
(x: 0.0004 ; yob: 14; aob: 50000; yod: 1; aod: 10000),
(x: 0.0005 ; yob: 20; aob: 60000; yod: 2; aod: 10000),
(x: 0.0006 ; yob: 28; aob: 80000; yod: 6; aod: 40000),
(x: 0.0007 ; yob: 40; aob: 120000; yod: 11; aod: 50000),
(x: 0.0008 ; yob: 52; aob: 120000; yod: 17; aod: 60000),
(x: 0.0009 ; yob: 70; aob: 180000; yod: 23; aod: 60000),
(x: 0.001 ; yob: 95; aob: 250000; yod: 29; aod: 60000),
(x: 0.0011 ; yob: 125; aob: 300000; yod: 35; aod: 60000),
(x: 0.0012 ; yob: 165; aob: 400000; yod: 43; aod: 80000),
(x: 0.0013 ; yob: 202; aob: 370000; yod: 54; aod: 110000),
(x: 0.0014 ; yob: 245; aob: 0; yod: 66; aod: 0))),,
(naz:'m^otek'; v0_N:-0.01; m_s:0.7; m_d:0.57; v0_T:0.01; dF:0.01;
T:((x: 0 ; yob: 0; aob: 0; yod: 0; aod: 0),
(x: 0.0001 ; yob: 0.325 ; aob: 3250; yod: 0.325; aod: 3250),
(x: 0.0002 ; yob: 0.65 ; aob: 3250; yod: 0.65 ; aod: 3250),
(x: 0.0003 ; yob: 0.975 ; aob: 3250; yod: 0.975; aod: 3250),
(x: 0.0004 ; yob: 1.3 ; aob: 3250; yod: 1.3 ; aod: 3250),
(x: 0.0005 ; yob: 1.625 ; aob: 3250; yod: 1.625; aod: 3250),
(x: 0.0006 ; yob: 1.95 ; aob: 3250; yod: 1.95 ; aod: 3250),
(x: 0.0007 ; yob: 2.275 ; aob: 3250; yod: 2.275; aod: 3250),
(x: 0.0008 ; yob: 2.6 ; aob: 3250; yod: 2.6 ; aod: 3250),
```

```

(x: 0.0009 ; yob: 2.925 ; aob: 3250; yod: 2.925; aod: 3250),
(x: 0.001 ; yob: 3.25 ; aob: 3250; yod: 3.25 ; aod: 3250),
(x: 0.0011 ; yob: 3.575 ; aob: 3250; yod: 3.575; aod: 3250),
(x: 0.0012 ; yob: 3.9 ; aob: 3250; yod: 3.9 ; aod: 3250),
(x: 0.0013 ; yob: 4.225 ; aob: 3250; yod: 4.225; aod: 3250),
(x: 0.0014 ; yob: 4.55 ; aob: 0; yod: 4.55 ; aod: 0))),

(naz:'chwytnik'; v0_N:-0.01; m_s:0.7; m_d:0.57; v0_T:0.01; dF:0.01;
T:((x: 0 ; yob: 0; aob: 0; yod: 0; aod: 0),
(x: 0.0001 ; yob: 0.325 ; aob: 3250; yod: 0.325; aod: 3250),
(x: 0.0002 ; yob: 0.65 ; aob: 3250; yod: 0.65 ; aod: 3250),
(x: 0.0003 ; yob: 0.975 ; aob: 3250; yod: 0.975; aod: 3250),
(x: 0.0004 ; yob: 1.3 ; aob: 3250; yod: 1.3 ; aod: 3250),
(x: 0.0005 ; yob: 1.625 ; aob: 3250; yod: 1.625; aod: 3250),
(x: 0.0006 ; yob: 1.95 ; aob: 3250; yod: 1.95 ; aod: 3250),
(x: 0.0007 ; yob: 2.275 ; aob: 3250; yod: 2.275; aod: 3250),
(x: 0.0008 ; yob: 2.6 ; aob: 3250; yod: 2.6 ; aod: 3250),
(x: 0.0009 ; yob: 2.925 ; aob: 3250; yod: 2.925; aod: 3250),
(x: 0.001 ; yob: 3.25 ; aob: 3250; yod: 3.25 ; aod: 3250),
(x: 0.0011 ; yob: 3.575 ; aob: 3250; yod: 3.575; aod: 3250),
(x: 0.0012 ; yob: 3.9 ; aob: 3250; yod: 3.9 ; aod: 3250),
(x: 0.0013 ; yob: 4.225 ; aob: 3250; yod: 4.225; aod: 3250),
(x: 0.0014 ; yob: 4.55 ; aob: 3250; yod: 4.55 ; aod: 3250))));

POPO : array[1..l_pop] of poprawki=
((tyt:'Plik parametr`w - ver.4.1'; ob:' '),
(tyt:'-----'; ob:' '),
(tyt:'pochyl.chwytnika'; ob:'P'; po:'a'; c:6; p:10;j:0.001; pop:0.01),
(tyt:'dno klaw.'; ob:'P'; po:'y'; c:0; p:15;j:0.0001;pop:0.0),
(tyt:'wys.pilota' ; ob:'P'; po:'r'; c:6; p:14;j:0.0001;pop:0.0005),

(tyt:'ogr.dkw.rep.' ; ob:'P'; po:'y'; c:0; p:10;j:0.0001;pop:0.001),
(tyt:'ogr.gorne dkw.' ; ob:'P'; po:'r'; c:1; p:25;j:0.0001;pop:0.0014),
(tyt:'spr@I.dkw.' ; ob:'SP'; po:'m0'; c:2; p:0; j:0.0001;pop:0.0),

(tyt:'ogr.lewe bij.' ; ob:'P'; po:'r'; c:3; p:12;j:0.0001;pop:0.0012),
(tyt:'dl. bijnika' ; ob:'P'; po:'r'; c:3; p:4; j:0.0001;pop:0.0027),
(tyt:'spr@I.bij.' ; ob:'SP'; po:'m0'; c:3; p:0; j:0.005; pop:0),
(tyt:'wys.pupki' ; ob:'P'; po:'y'; c:0; p:7; j:0.0001;pop:0.0065),

(tyt:'odl.chwytnika' ; ob:'P'; po:'r'; c:6; p:6; j:0.0001;pop:0.0013),

(tyt:'po . yłeczki' ; ob:'P'; po:'y'; c:5; p:14;j:0.0001;pop:-0.0022),
(tyt:'dl.pr@tu tlum.' ; ob:'P'; po:'y'; c:5; p:8; j:0.0001;pop:-0.0099),
(tyt:'-----'; ob:' '),
(tyt:'obr`t figury' ; ob:'P'; po:'a'; c:1; p:2; j:0.001; pop:0),
(tyt:'obr`t dkw.rep.' ; ob:'P'; po:'a'; c:2; p:2; j:0.001; pop:0.043),
(tyt:'obr`t bijnika' ; ob:'P'; po:'a'; c:3; p:2; j:0.001; pop:0),
(tyt:'obr`t m otka' ; ob:'P'; po:'a'; c:4; p:2; j:0.001; pop:0),
(tyt:'obr`t t umika' ; ob:'P'; po:'a'; c:5; p:2; j:0.001; pop:0),
(tyt:'obr`t klawisza' ; ob:'P'; po:'a'; c:6; p:2; j:0.001; pop:-0.001));

var
i,j,n : integer;
p_pkt;
ch : char;
pop : poprawki;
POP : array[1..l_pop] of poprawki;
begin

Assign(Fmat,File_mat);
rewrite(Fmat);
writeln('Materialy');
for i:=1 to l_mat do
  write(Fmat,MAT0[i]);
writeln('Liczba rekord`w : ',filesize(Fmat));
close(Fmat);

for i:=1 to l_pop do
  POP[i]:=POP0[i];
writeln('Zmienic wartosci poprawek? (t/n)');


```

```
readln(ch);
if (ch='n') then
begin
  Assign(Fpop,File_pop);
  Reset(Fpop);
  for i:=1 to l_pop do
    begin
      read(Fpop,popp);
      POP[i].pop:=popp.pop;
    end;
  close(Fpop);
end;

Assign(Fpop,File_pop);
Rewrite(Fpop);
Writeln('Parametry');
for i:=1 to l_pop do
  Write(Fpop,POP[i]);
writeln('Liczba rekordów : ',filesize(Fpop));
close(Fpop);
readln;
end.
```

Spis rysunków.

Rys.1.	Schemat budowy mechaniki fortepianu marki Stainway.	12
Rys.2.	Schemat działania mechaniki fortepianu.	13
Rys.3.	Uproszczony schemat mechanizmu młoteczkowego oraz występujące w nim kontakty. K1-pilot-stopka; K2 - bijnik-pupka; K3 – bijnik-ogranicznik lewy; K4 – bijnik-ogranicznik prawy (w dźwigni rep.); K5 – bijnik (powierzchnia tylnej)-baryłka; K6 –bijnik (powierzchnia górna-główka)-baryłka; K7 – baryłka-dźwignia rep.; K8 - dźwignia rep.-ogranicznik; K9,10 - dźwignia rep.-ograniczniki figury (górny i dolny); K11 -młotek-struna; K12 - młotek-chwytnik; K13 - tłumik-struna; K14 – tłumik-ogranicznik; K15 - klawisz-łyżka tłumika; K16 - klawisz-ogranicznik (oparcie klawisza); K17 – klawisz-dno klawiatury;	14
Rys.4.	Przypisanie na stałe punktów kontaktu.	15
Rys.5.	Struktura programu.	17
Rys.6.	Struktura procedury <i>Rów Ruchu</i>.	18
Rys.7.	Przykładowe siły działające na ciało 1; Q – siła grawitacji przyłożona w środku masy; r_m – wektor łączący oś obrotu ciała z środkiem masy; F_k – siła reakcji działająca w punkcie kontaktu z ciałem 2; r_k – wektor łączący oś obrotu ciała z umownym punktem kontaktu; M-suma momentu tłumiącego i momentu pochodzącego od sprężynek.	19
Rys.8.	Przykładowe siły działające na łańcuch ciał 1 i 2 (konwencja oznaczeń ta sama, co na rys.5.; drugi indeks odpowiada numerowi ciała).	20
Rys.9.	Uwolnienie od więzów.	21
Rys.10.	Przykładowy łańcuch 3-elementowy (Konwencja oznaczeń ta sama, co na rys.5. Przy punktach podano ich numery poprzedzone numerami ciał).	28
Rys.11.	Przykładowy układ dwóch odcinków.	29
Rys.12.	Działanie sprężynek (linią ciągłą narysowane są ciała w położeniu początkowym - $\alpha_1=\alpha_2=0$).	33
Rys.13.	Przykład zamodelowania kontaktu bijnika z baryłką.	35
Rys.14.	Sposób budowania kontaktu.	35
Rys.15.	Rodzaje kontaktu (opisane zmienną st).	36
Rys.16.	Składowe prędkości penetracji normalnej i stycznej dla okręgu i odcinka.	36
Rys.17.	Układ klocek – podłożę.	40
Rys.18.	Porównanie udziału siły wymuszenia działającej na barłkę ($F_{1,2}$) i bijnik ($F_{1,1}$).	41
Rys.19.	Rozkład sił wymuszających pomiędzy kontakty.	42
Rys.20.	Stanowisko badawcze do pomiarów właściwości sprząstych materiałów.	56
Rys.21.	Schemat blokowy czujnika ADXL210.	56
Rys.22.	Idea działania akcelerometru	57
Rys.23.	Różne możliwości przeprowadzenia pomiarów.	58
Rys.24.	Pomiar sprężystości chwytnika.	59
Rys.25.	Orientacje czujnika przy kalibracji.	59
Rys.26.	Wpływ miejsca nakluwania młotka na barwę dźwięku.	67
Rys.27.	Schemat stanowiska badawczego	70
Rys.28.	Schemat działania czujnika tensometrycznego podłączonego do półmostka.	71
Rys.29.	Przykładowa klatka zdjuciowa z widocznymi markerami.	73
Rys.30.	Ruch wahadłowy ciała.	76
Rys.31.	Schemat modelu uproszczonego.	84

Spis tabel.

Tab.1.	Przykładowa mapa punktów.	30
------------------------	---	----

<u>Tab.2.</u>	<u>Przykładowa tablica P porządkująca geometrię.</u>	30
<u>Tab.3.</u>	<u>Przykładowa lista poprawek.</u>	30
<u>Tab.4.</u>	<u>Przykładowa tablica ST dla układu z rys.10. Kontakt nr 1 opisuje przenikanie ciała 1 i 4; nK=2 – ciała 2 i 5; nK=3 – ciała 3 i 6;</u>	32
<u>Tab.5.</u>	<u>Porównanie wyników dla różnej dokładności całkowania.</u>	77

Spis wykresów.

<u>Wyk.1.</u>	<u>Przykładowa symulacja tłumienia oscylacji figury poru.</u>	34
<u>Wyk.2.</u>	<u>Przykładowa pętla histerezy siły reakcji ciała.</u>	38
<u>Wyk.3.</u>	<u>Współczynnik wagowy.</u>	38
<u>Wyk.4.</u>	<u>Funkcja wagowa tarcia.</u>	40
<u>Wyk.5.</u>	<u>Funkcja tarcia statycznego.</u>	40
<u>Wyk.6.</u>	<u>Przedział całkowania.</u>	47
<u>Wyk.7.</u>	<u>Możliwe przebiegi przyspieszeń podczas zderzeń</u>	58
<u>Wyk.8.</u>	<u>Liniowe charakterystyki akcelerometrów.</u>	59
<u>Wyk.9.</u>	<u>Przykładowy przebieg przyspieszeń dla baryłki.</u>	60
<u>Wyk.10.</u>	<u>Szacowanie czasu kontaktu.</u>	60
<u>Wyk.11.</u>	<u>Przykładowy przebieg prędkości dla baryłki.</u>	61
<u>Wyk.12.</u>	<u>Przykładowy przebieg przemieszczeń baryłki w przedziale czasu $t_0 - t_1$.</u>	61
<u>Wyk.13.</u>	<u>Przykładowa pętla histerezy siły reakcji od głębokości penetracji dla baryłki</u>	62
<u>Wyk.14.</u>	<u>Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla baryłki</u>	62
<u>Wyk.15.</u>	<u>Serie danych dla obciążenia i odciążenia oraz zasymulowane pętle histerez dla baryłki.</u>	63
<u>Wyk.16.</u>	<u>Przykładowy przebieg siły reakcji od penetracji w kontakcie 1 - pilot-stopka.</u>	64
<u>Wyk.17.</u>	<u>Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla filcu dna klawiatury. Kat poczatkowego wychylenia wahadła $\alpha_0=5^\circ-15^\circ$.</u>	65
<u>Wyk.18.</u>	<u>Serie danych dla obciążenia i odciążenia oraz zasymulowana pętla histerezy dla filcu dna klawiatury.</u>	65
<u>Wyk.19.</u>	<u>Rodzina Pętli histerezy siły reakcji od głębokości penetracji dla filcu stopki. Kat $\alpha_0=8^\circ-15^\circ$.</u>	66
<u>Wyk.20.</u>	<u>Serie danych dla obciążenia i odciążenia oraz zasymulowana pętla histerezy dla filcu stopki</u>	66
<u>Wyk.21.</u>	<u>Wpływ twardości młotka na widmo akustyczne dźwięku.</u>	68
<u>Wyk.22.</u>	<u>Przykładowe czasy kontaktu młotka ze struną dla dźwięku C⁴ w zależności od jego głośności [3].</u>	68
<u>Wyk.23.</u>	<u>Rodzina pętli histerezy dla młoteczka i jej aproksymacja liniowa.</u>	69
<u>Wyk.24.</u>	<u>Charakterystyka siłomierza.</u>	71
<u>Wyk.25.</u>	<u>Filtrowanie sygnału z siłomierza.</u>	72
<u>Wyk.26.</u>	<u>Zestawienie dwóch serii danych zebranych przez kamery dla przemieszczeń młoteczka</u>	73
<u>Wyk.27.</u>	<u>Porównanie wyników z kamery i czujnika dla przemieszczeń młotka.</u>	74
<u>Wyk.28.</u>	<u>Porównanie wyników z kamery i czujnika dla przemieszczeń klawisza.</u>	74
<u>Wyk.29.</u>	<u>Względne różnice w wynikach z kamery i z czujników dla przemieszczeń klawisza i młotka.</u>	75
<u>Wyk.30.</u>	<u>Przebiegi sił wymuszenia dla dwóch serii danych.</u>	76
<u>Wyk.31.</u>	<u>Porównanie prędkości młoteczka dla różnej dokładności całkowania (seria 1. i 2.).</u>	77
<u>Wyk.32.</u>	<u>Porównanie prędkości młoteczka dla różnej dokładności całkowania (seria 2. i 3.).</u>	78
<u>Wyk.33.</u>	<u>Wykorzystany w symulacji przebieg siły wymuszenia działającej na klawisz.</u>	78
<u>Wyk.34.</u>	<u>Porównanie wyników symulacji i doświadczenia dla przemieszczeń klawisza.</u>	79
<u>Wyk.35.</u>	<u>Porównanie wyników symulacji i doświadczenia dla przemieszczeń młotka.</u>	79
<u>Wyk.36.</u>	<u>Przebieg procesu wyhamowania młotka przez chwytnik.</u>	80
<u>Wyk.37.</u>	<u>Porównanie wyników symulacji i doświadczenia dla prędkości klawisza.</u>	81
<u>Wyk.38.</u>	<u>Porównanie wyników symulacji i doświadczenia dla prędkości młotka.</u>	82

<u>Wyk.39.</u>	<u>Porównanie wyników symulacji i doświadczenia dla przyspieszeń młotka przy uderzeniu w strunę.</u>	
		82
<u>Wyk.40.</u>	<u>Porównanie wyników symulacji i doświadczenia dla prędkości młotka przy podniesieniu się klawisza.</u>	
		83
<u>Wyk.41.</u>	<u>Porównanie położen młotka uzyskanych dla modelu "dokładnego" i uproszczonego.</u>	
		88
<u>Wyk.42.</u>	<u>Porównanie prędkości młotka uzyskanych dla modelu "dokładnego" i uproszczonego.</u>	
		88
<u>Wyk.43.</u>	<u>Porównanie sił wymuszających uzyskanych dla modelu "dokładnego" i uproszczonego.</u>	
		88

Spis treści.

<u>1.</u>	<u>Wstęp.</u>	3
<u>2.</u>	<u>Historia fortepianu.</u>	5
<u>3.</u>	<u>Model matematyczny.</u>	10
3.1.	<u>Budowa mechanizmu młoteczkowego</u>	11
3.2.	<u>Założenia symulacji</u>	14
3.3.	<u>Budowa algorytmu</u>	16
3.4.	<u>Równania ruchu</u>	19
3.5.	<u>Obliczanie położenia i prędkości punktów we wsp. XY.</u>	28
3.6.	<u>Momenty sprężynek i tłumienia.</u>	33
3.7.	<u>Siły wzajemnego oddziaływanie ciał (naciski i tarcie).</u>	34
3.8.	<u>Rozwiązywanie równań ruchu dla zespołu figury.</u>	44
3.9.	<u>Całkowanie równań ruchu.</u>	46
3.10.	<u>Regulacja mechanizmu.</u>	52
<u>4.</u>	<u>Pomiary doświadczalne.</u>	55
4.1.	<u>Pomiary sprężystości filców i skóry</u>	55
4.2.	<u>Pomiary charakterystyk dynamicznych mechaniki fortepianu.</u>	70
<u>5.</u>	<u>Weryfikacja symulacji.</u>	77
<u>6.</u>	<u>Model uproszczony</u>	84
<u>7.</u>	<u>Podsumowanie.</u>	90
<u>Bibliografia :</u>		91
<u>Unit Dane_4;</u>		92
<u>Unit Grafika4;</u>		98
<u>Unit Pr_num4;</u>		106
<u>Program Piano4;</u>		112
<u>Program Geom_4;</u>		127
<u>Program Matpop_4;</u>		133
<u>Spis rysunków.</u>		136
<u>Spis tabel.</u>		136
<u>Spis wykresów.</u>		137
<u>Spis treści.</u>		138